

S I N C L A I R

Every month £1.75 June 1989

4 to 7 9 10 14 16 18 20
22 26 34 38 BACK PLANE AND COL
39 OUTER BACK COVER.

QL WORLD

DO-IT-YOURSELF ARCHIVING

FILE TRANSFERS

THE NEW FLASHBACK



Editor
Helen Armstrong

Chief Sub Editor
Harold Mayes MBE

Production Manager
Nick Fry

Designer
Chris Winch

Advertising Sales
Paul Cave

Magazine Services
Sheila Baker

Advertising Production
Michelle Evans

Publisher
Perry Trevers

*Publishing and
Commercial Director*
Paul Coster

Financial Director
Brendan McGrath

Chief Executive
Richard Hease

Microdrive Exchange
089 283 4783/2952
(2 lines)

Sinclair QL World
Greencoat House
Francis Street
London SW1 1DG
Telephone 01-834 1717
Fax 01-828 0270
Telex 9419564 FOCUS G
ISSN 026806X

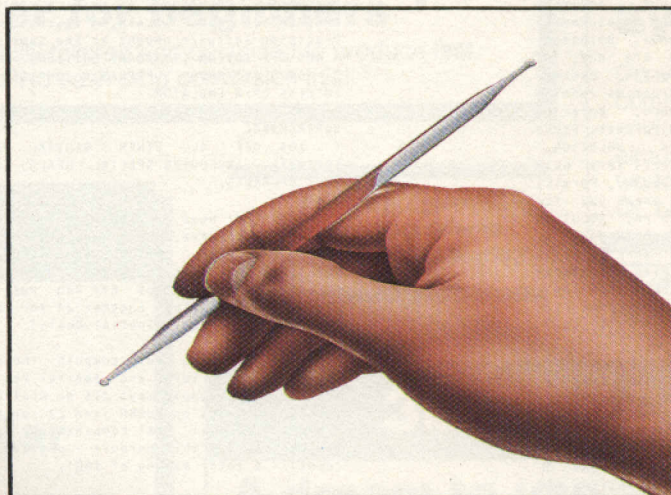
Unfortunately, we are no longer able to answer enquiries made by telephone. If you have any comments or difficulties, please write to The Editor, Open Channel, Trouble Shooter, or Psion Solutions. We will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies. Back issues are available from the publisher, price £2 U.K., £2.75 Europe. Overseas rates on request. Please telephone 089 283 4783 to check availability. Published by Focus Magazines Ltd., London. Distributed by S M Distribution, Streatham, London SW1.

Typesetting by Adtec Typographics, Stanford-le-Hope, Essex.
Tel: 0375 360967.
Printing by Southernprint Ltd.
© COPYRIGHT SINCLAIR QL WORLD — 1989

CONTENTS

■ ■ JUNE 1989

- 9 **QL SCENE ● Microdrive threat**
- 10 **OPEN CHANNEL ● Tilt and win . . .**
- 11 **SUBSCRIPTION INFORMATION**
- 14 **DIY ARCHIVING ● The first in a series on using Archive**
- 16 **SOFTWARE FILE ● QD file editor**
- 18 **FLASHBACK TO THE FUTURE ● The update reviewed**
- 22 **SUPERBASIC ● Part 4 of the calculator project**
- 26 **TRANSFERS WITHOUT TEARS ● File swapping — the options**
- 34 **DIY TOOLKIT ● Pin down a pixel**
- 38 **QUANTA SHOW ● High jinks at Northampton**
- 39 **PROGRAM OF THE MONTH ● Crazy Cards**
- 48 **MICRODRIVE EXCHANGE ● Football Manager — volunteers?**



NEXT MONTH

FLASHBACK — THE MODULES

Bryan Davies looks at the modules which accompany the recent Flashback update.

OF MICE AND ART

Old masters and young Painters seen through Ron Massey's critical eye.

Save Money Save Time

Subscriptions available from: TIL, PO Box 74, Paddock Wood, Tonbridge, Kent, TN12 6DW.
1 year's subscription: £15 U.K. £30 surface mail — Europe and the rest of the world. Add £5 for air mail plus £10 overseas.
Tel: 089 283 4783

QL

S C E N E

Microcassette manufacturers may quit QL

Signs have appeared that the much-maligned Sinclair micro-cassette may at last be an endangered species.

Ablex, the company which has made the tape cartridges since the start, reports that supplies of suitable magnetic tape from German comms giant BASF may dry up.

Even if stocks are found, says production manager David MacSorley, Ablex will not make microcassettes after 1989.

Says MacSorley, "We are waiting for alternative samples from BASF, which we will have to test." If an alternative is located, Ablex "would be interested in carrying it for the rest of the year." He gave the declining market as the reason.

MacSorley was pessimistic about another company taking over production. *QL World* has not yet had a response from Sinclair Research but hopes to report further next month.

QL BB breaks away

The QL bulletin board planned by David Park has now parted company with Quanta and will be setting up at a new site and number soon. "We have moved and we will have to wait for a new line to be installed,"

says the sysop. "We had more than 400 enquiries when the board was announced. Please ask your readers not to ring the original number and we will release the new number as soon as we have it."

Please will reader/contributor John Banks contact us? Letters to his college address have been returned marked 'gone away'.

Check the cheque

American software publisher Sharps has released a *Check-book Manager* program — named in the U.S. style. The program costs £13.95 which includes airmail postage to the U.K. and Europe. *QL World* expects to review the program soon.

Sharps, Box 326, Mechanicsville, VA 32111, U.S.A.

Subscription breather

QL World publisher Focus Magazines has modified its plan to convert to a subscription-only base to give readers, the news trade and the company more time to acclimatise.

The large number of new subscriptions arriving coincided with postal distribution

problems, while many "occasional" readers remain unaware of the planned changes.

The number of magazines available from newagents will be rolled off gradually up to the end of 1989, allowing the news to circulate, and the subscription base to build up in stages.

Banker upgrade

DJW Software, publisher of *Home Banker Plus*, is issuing an upgrade version of the program, *Home Banker Plus V3.1*. Features added since the program was reviewed in the March issue of *QL World* include single-key selection on the main and standing order menus, up to 14 accounts on each file, improved account balancing and standing order transactions, and a new owners' manual.

Home Banker Plus V3.1 is supplied on a 3.5in. disc for

QLs with a minimum of 256K memory expansion. DJW Software will be contacting existing customers with information on how to upgrade copies of *Home Banker* to the current version. The new package costs £19.95 and is available from Sector Software.

DJW Software, 11 Pound Close, Bramley, Basingstoke, Hampshire RG26 5BL.

Sector Software, 39 Wray Crescent, Unles Walton, Leyland, Lancashire PR5 3NA. Tel: 0772 454328.

Show scene thrives for QL

Taurus Computer Systems, promoter of the Alternative Micro Shows, reported an attendance of more than 2,000 at its spring show at London's New Horticultural Halls.

Our observer reports that the overall scope of the show was "much smaller than a Microfair." QL dealers quoted as attending include Quanta, Digital Precision, Sector Software, Care Electronics, QS Sub, PDQL, Schön PC and Miracle Systems, and there seems to have been a good trade in boxed QLs.

Taurus was formerly Emsoft Ltd, which started life as a

support group for the Tatung Einstein and has branched out into promoting shows for specialist computers.

Taurus address is 6 St. Ives Close, Kesgrave, Ipswich, Suffolk IP5 7LT. Tel: 0473 622789. Another Alternative Micro Show is planned for Saturday, November 11 at Bingley Hall, Stafford. A colour brochure is available.

QL World assessment of Alternative Micro Shows is that, once established, they have the potential to attract large attendances to central show sites which might not otherwise be accessible to indi-

vidual computer shows. At present, they are of moderate interest to QL users. We hope to have advance information of companies attending the November show.

Meanwhile, David Batty of Sector Software reports attendance of more than 1,000 QL users at his Northern Sinclair Show in Manchester, with a great deal of local interest, especially from users who would not normally make the journey south for Microfairs. Another Manchester show is planned in three months. Contact Sector Software, 0772 454328, for information.

OPEN CHANNEL

Open Channel is where you have the opportunity to voice your opinions in *Sinclair QL World*. Whether you want to ask for help with a technical problem, provide somebody

with the answer, or just sound off about something which bothers you, write to: Open Channel, Sinclair QL World, Greencoat House, Francis Street, London SW1 1DG.

Two tips

I have two tips for QLers. By accident I found that a three-quarter-inch board placed under the back of the QL, with the QL rear support legs in place, will tilt the QL forward for a more comfortable keying position and, if the board is secured, will locate the QL firmly. I have a SuperQboard plus 512K RAM extension with twin 3.5in. floppy drives and PAR printer driver, internal SuperQmouse, Qram and CPMulator in the ROM socket. That is almost a full house and I get the occasional crash but not since this new arrange-

ment. Presumably the increased angle helps through ventilation to keep components below crash temperature.

Every day I use Abacus to enter changes in share prices and as a matter of course preload F1 to F2 with Altkey Abacus instruction sequences while booting in SuperBasic. With the multi-tasking facilities of Qram, it is always easy to nip back into SuperBasic to add an Altkey sequence and then return to Psion to use it for repetitive commands. One boot-loaded Altkey is:

```
ALKEY CHR$(240), CHR$(236), CHR$(240)&'lshrs',
```

```
CHR$(216), CHR$(216),  
CHR$(240)&'wh',  
CHR$(248)  
&'A26',
```

which will remove prompts, load 'shrs', drop the cursor two lines, split the window horizontally, GOTO A26 — bring a section of the spreadsheet with top left corner at A26 on to the screen — with the touch of ALT/F3.

Many QL users may not realise the time-saving possibilities.

**W.T.M. Lawrence,
Bexley,
Kent.**

subscription enquiries but full details appear on the Contents page every month.

Crystal call

I would like to take this opportunity to say what a great magazine *QL World* is. Would you ask any readers who live in the Penge, Crystal Palace or Beckenham areas to write to me so that I can start a local QL club?

**Brian Dickson,
67 Queen Adelaide Road,
Penge, London SW20**

Detective

I am trying to obtain a copy of *QL SuperBasic, The Definitive Handbook*, written by Jan Jones. The publisher McGraw-Hill advises that the book is out of print and not in stock.

If anyone can advise me where I can get a copy I shall be grateful.

**C. J. Garnett,
5 Park Road,
Coombs Park,
Coleford,
Gloucestershire.**

All the fax

Thank you for the information about ABC Elektronik. If the Instant Access panel is to be of any help, someone at Focus needs to perform an update on it. You should also include fax numbers for all firms which have one.

**Don Atkins,
Sydney.**

Editor's reply: The latest update of Instant Access appears in this issue. Instant Access is a quick reference for contacts and is not intended to be a substitute for the monthly advertisers' index or advertisements. It covers established QL supporters and the last known numbers of a few companies no longer supporting the QL. Space is restricted and very few private users have fax. Check the companies advertising for further details.

For example, the ABC Elektronik advertisement on page 13 of the April issue gives the fax number. Incidentally, Instant Access now contains up-to-date information about QL World

Would any of your readers be interested in translating a suite of programs written in Basic for the ZX Spectrum to run on a QL? I can supply either Microdrives or listings and would agree to a reasonable fee.

**Colin Hodson,
Hodson Rivers,
2 Ridgemount Street,
London WC1E 7AA.**

Editor's comment: Any interested programmers should contact Hodson directly with their offers.

Editor's notebook

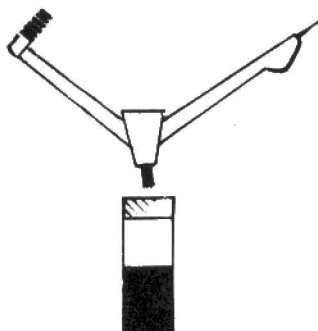
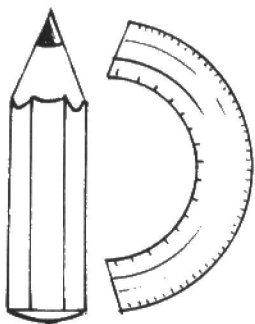
Is the "QL in decline", or is it not? How do you define "decline", anyway?

This speculation has been going on for as long as I can remember. If you define decline as "fewer people, less money", there is less to go round. If you define it as "less up-to-date, less useful", this is not the case at all. QL hardware and software is advancing almost weekly, and the QL's unusual and advanced capabilities remain ahead of the field.

Reconciling the lower 'popular' demand for the QL and the continued serious demand is setting QL traders some riddles.

Our own riddle is how to reduced unproductive distribution costs. The prescription is to rely on subscription; however, we realise that everyone needs more time to adapt to the change. So, although the cover price has to be raised, the price of subscription is to stay down, and the magazine will still be available 'one off' from newsagents till the end of the year.

The second riddle is that of the microcassettes. When Ablex ceases to produce them, will anyone else be able, willing and permitted to carry on making them. Some users talk of disc drives, but this sounds like "let them eat cake" to me. Microcassettes are maligned, but still popular. We will keep you posted.



ARC

John Davis
presents the first
of an occasional
series on
Archive
programming.
This month he
relates Archive
to Basic.

I am a coarse programmer and this article is written for other coarse programmers. Six years ago I knew nothing about programming but I had a time-consuming annual task which seemed to cry for a computer. It involved the preparation of an annual rota for some 50 magistrates. The first home computers appeared but I could get no indication as to whether they had the power for the task I had in mind and they were too costly to buy on spec. Then Clive Sinclair produced the ZX-81 and I was hooked.

I set about learning programming and writing my rota program at the same time. As I saw a possible application for something I had learned, I tried it and kept at it until it worked. I learned rapidly, from hard experience, to save it at least twice and so I am now obsessive about back-up copies.

It is for people who tend to program freehand I am writing. I hope there may be things of some value to people with a formal background in programming. On the other hand I hope that comparative novices may find it possible to work their way into Archive through these articles.

Eventually I was in a position to move to a QL. Before starting the conversion of my program I looked in the Archive section but was deterred by the lack of line numbers and the elegant layout of the procedures. What were procedures anyway?

As a result I set about converting my program from ZX-81 Basic into SuperBasic. I learned a little about procedures and, eventually, when I looked again at Archive I recognised it for what it was, a special dialect of Basic.

If you wish to print a number of values in a column, right-justified in the conventional way for listing figures with the units above the units, the 10s above the 10s and so on, for simplicity they are all whole numbers in

the range 1 to 9999. In Basic you could write a sub-routine:

```
1000 LET Y$ = str(Y)
1010 LET Spaces$ = " "
1020 LET Spaces$ = spaces$ (1 to
len(spaces$)-len(Y$))+Y$
1030 PRINT TAB X; Spaces$
1040 RETURN
```

Having written this you could then include in the main program:

```
10 LET X = 10
20 FOR a = 1 to 10
30 LET Y = NUM(a)
40 GOSUB 1000
50 NEXT a
```

The result would be to print the contents of the numeric array NUM in a four-wide character column starting 10 from the margin. Effectively that sub-routine is a PROCEDURE and the values X and Y are its PARAMETERS.

In SuperBasic the subroutine could be re-written as a PROCEDURE in this form:

```
1000 DEFine PROCedure justify (x,y)
1010 spaces$ = fills$(" ",5-(len(str(y))&
str(y))
1020 PRINT TAB x;spaces$
1030 END DEFine
```

In the main program lines 10, 30 and 40 are replaced by:

```
30 justify 10,num(a)
```

Procedures do things and need not always have any parameter passed to them, e.g.:

```
1000 DEFine PROCedure home
1010 AT 0,0
1020 END DEFine
```

This has no PARAMETERS but it will, if the word 'home' is typed-in as a command or is included in a program, invariably return the print position to the top left-hand corner of the screen at a saving of two key-strokes every time it is used.

By contrast, FUNCTIONS must always have at least one PARAMETER, based on the value of which it RETURNS another value.

If my 'justify' procedure were to be re-written as a FUNCTION, the parameter, x would probably be omitted and line 1030 would have to be re-written:

```
1030 RETURN spaces$
```

It would be called in the main program by an amended line 30:

```
30 PRINT TAB x;justify(y)
```

The parameter x would now need to be replaced by a program variable or its value could be substituted in line 30.

Once you have used procedures for a time you realise that GOSUB is redundant. For the coarse programmer GOTO still has its attractions but I have been persuaded that there are good reasons, not merely programming fashion, to eschew its use. Even when used completely in a single procedure it sacrifices a major advantage of procedures, namely speed. Use of GOTO involves the machine going right through the program looking for that line number, whereas it has a much shorter index of the start addresses of procedures and functions.

Once so persuaded, I found that it was not so difficult to avoid GOTO as well. Once I had been living without GOTO as well as GOSUB for a time I wondered who needs line numbers? Once you realise that you are ready to approach Archive.

For anyone who started on the ZX-81, moving from SuperBasic to Archive is a little like coming home for the following reasons:

a. To send to the printer you use LPRINT rather than OPEN#4,ser1ehc: PRINT#4:CLOSE#4. You can divert data intended for the printer to a "000 lis" file by use of the commands SPOOLON and SPOOLOFF. e.g.:

```
SPOOLON "fred"
LPRINT fred$; tab 30;smith$
LPRINT etc. etc
SPOOLOFF
```

will produce a file on your data device — mdv2 00 if you have not re-configured Archive — called "fred00 lis" which will contain the value of fred\$ and smith\$ on one line followed by the other values LPRINTed.

You can check your layouts without wasting paper by preceding the LPRINTs with SPOOLON SCREEN. This has one important limitation in that, if you have attempted to force a formfeed by a line LPRINT chr(0)+chr(12) it will stop with a report of "i/o error"; the screen will not accept nonprintable characters. You can avoid this problem by writing a Procedure

CHIVING

```
FF:
proc FF
  lprint chr(0)+chr(12)
endproc
```

and including the procedure call "FF" in the document layout. If wishing to test the layout on the screen you can merge in a dummy proc FF for the duration of the test and then merge back the proper one. The dummy procedure would read:

```
proc FF
  endproc
b. As you will have seen in a, strings are concatenated by "+" and not "&". That will be familiar to those who started on the ZX-81. Other differences are:
```

"chr(n)" not "chr\$(n)" likewise "str(n)". There is no coercion so "LET a = a\$(3 to 4)" is unacceptable.

Beginners are reminded that concatenated means chained, e.g., "fat" + "head" = fathead

c. Use of LET is mandatory, e.g., "x = 10" will return an error.

d. Commands and functions appear in lower-case so, if you wish to check that you have not used one of these as a procedure call in place of a similarly-named procedure, use upper-case for procedure names and check that they are not changed to lower-case when they appear on the screen.

Commands are 'reserved words' and you may not use them either as procedure names or variable names, even with the addition of '\$' for strings. The line editor will check that you have not used a reserved word as a procedure name but it will let you use it as a variable name but this will cause the program to halt with an error when it comes to the line. Function names, without the brackets used to contain the parameters, can be used as variable or procedure names. So the line editor will NOT accept:

```
proc all — (command used as a procedure name)
let a$ = fred — (no closing quote).
```

but it will accept the following, although they will not run:

```
let all = 12
(command used as variable name)
```

```
let all$ = "fred" (the same)
let a$ = 12 (mixed data types)
let a = "fred" (the same)
The following is acceptable both on input and when run:
```

```
let date$ = date(1) (function name used for variable)
```

e. DEFINE programs in Archive consist entirely of a series of procedures. There are a number of pre-defined functions but there is no provision for user-defined functions. For that reason the keyword DEFINE is not used. An example of pre-defined function which exists in Archive but not in SuperBasic is "dec(n,d,w)". This is a sophisticated version of my example "justify". It takes three parameters:

n. being the value to be printed
d. being the number of decimal places
w. being the width of the field in which it is to be printed

So, if price has been calculated as 2.736, the statement:

```
print "£";dec(price,2,5)
```

will produce the output "£2.74"

f. TRACE. Since there are no line numbers, when an Archive program stops on an error it displays the procedure name and the line. So if, in a procedure called "bill" the foregoing print example were used without price having been declared, it would stop with the error number for undefined variable after a line which showed:

```
bill print "£";dec(price,2,5)
```

The toggled command "trac" enables the programmer to follow the operation of his program. When trace is on, each line is displayed in the foregoing form and so it is possible, when debugging, to follow the flow of the program.

g. Archive has an ERROR provision which enables a program to deal with error situations without halting. I will show how this can be used to close an indeterminate number of open files and then continue the program, without operator intervention.

h. Loops Basic had just "For/Next" loops; Superbasic added "Repeat/End Repeat" loops with exit. Archive has "While/Endwhile" and "All/Endall" loops.

"While" loops are a little like repeat loops with the exit condition contained in the line which marks the beginning of the loop. They may be made to operate as for/next loops by the inclusion of a counter, e.g.:

```
let count = 1
while count<10
  print tab 10; count
  let count = count+1
endwhile
```

All/Endall provides the fastest way to search through a database file but must not be used where the records are to be updated in the loop.

i. Archive is designed to manipulate database files. For those used to Basic or SuperBasic, a database file can be regarded as an expandable array capable of holding both string and numeric data. Another image would be that of the card index with pre-printed cards requiring specific ordered data. The box of cards is the 'File', each card is a 'Record' and each piece of required data, e.g., 'Name' or 'Date of Birth' is a 'field'. A database can consist of one or more such files.

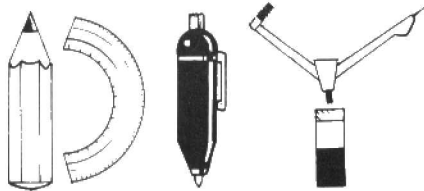
To start editing, load Archive as you would Quill, i.e., re-set the machine, place the program medium in mdv100 or flp100 and press F1 or F2. The screen will display the usual prompts at the top and a broad line near the bottom. The cursor will be in the 'command area' below the line.

If you press F3 a list of commands will appear in the centre of the prompts area but, unlike the other three Psion programs, pressing the initial letter of one of these commands has no effect, other than to make that letter appear at the cursor position. As in SuperBasic the whole word has to be typed and then <Enter> pressed. Certain commands will have a familiar look but will operate slightly differently.

Dir — displays the directory of a device
Load — loads a program
Save — saves a program

Unlike Basic or SuperBasic, the device and or file names have to be in double quotes. Nevertheless, if you type the comm and press <ENTER> the interpreter will offer a pair of quotation marks with the cursor between.

Continued on page 16



Sload and Ssave which load and save screens respectively work in the same way but somewhat illogically. The following two commands:

Save object and
Load object

which speed loading and saving by condensing the file into machine code, do not have the facility of offering the quotes. If you forget this you have no choice but to type everything again because, unlike SuperBasic, Archive does not return a 'bad line' for correction; you just have to start again.

Type "Edit" and <ENTER>. A thick vertical line will appear about 10 columns in from the left of the screen. In the command area there will have appeared the word Proc with the cursor just to its right. Type "test;t,m" <ENTER>. The word "test" will appear to the left of the vertical line and the following will appear to the right: As you type and <ENTER> the following lines they will be inserted in sequence between proc and endproc:

```
if m = <ENTER>
mode <ENTER>
endif <ENTER>
if t <ENTER>
trace <ENTER>
endif <ENTER>
rem here insert procedure call to be tested
<ENTER>
if t <ENTER>
trace <ENTER>
endif <ENTER>
mode 1 <ENTER>
```

On completing this the procedure will read:

```
proc;t,m
if m = 0
mode 0
endif
if t
trace
endif
rem here insert the procedure call to be
tested if t
trace
endif
mode 1
endproc
```

If any lines require to be amended, use the up cursor to move up the highlight to the line concerned and press F5. The line concerned will be brought down to the command area where it can be edited. If a

line has been omitted, use the cursor keys to highlight the line before it and press F4, type the missing line and <ENTER>. To leave this insertion mode press <ESC>. When you are satisfied with the procedure press <ESC> to leave editing mode. On return from the editing mode type:

```
save <ENTER> (quotes will appear
around cursor)
testrig <ENTER> (this should appear
between the quotes)
```

You will then have saved this one procedure as "testrig prg" on the drive configured as the data device. Test it by typing:

```
Test;1,1 <ENTER>
Test;1,0 <ENTER> and
Test;0,0 <ENTER>
```

You have now created and tested a test rig which can be merged into any future program you are debugging. A procedure call with all necessary parameters can be edited into this procedure, which can then be used repeatedly with or without trace to test other procedures in the new program. Use of Mode 0 makes the whole screen area available to display the 'trace'; otherwise only the command area is used, which makes it a little difficult to follow.

The next part will deal with copying and opening files.

SOFTWARE FILE

QD is a text editor, the only one to work under the QJump pointer environment, and it is a smart piece of programming. The review copy was on a 3.5in. disc. The documentation was a prototype, being translated from German to English by the author, and was in the form of a small black hardback ringbinder. Whether this is an example of the proper manual which will be produced for general sale, I cannot say but I hope it will be, as it lends an added quality to the product. Again, though, QD has a quality of its own which makes it worth the price anyway.

The documentation gave an introduction to QD and explained the concepts. It dealt with the menus in sections, each being separated with a coloured page card. I found the documentation clearly printed, well laid-out and pleasant to read.

I tested QD for a period of days on my QL, which is expanded with Trump Card to give me 896K of memory and had no problems. I used the Toolkit II command RES_128 to bring the QL back to the state of being unexpanded and tried to load QD but had no luck.

You do not have to use the standard boot to load QD. You could EXEC it in but first you must install two files PTR_GEN, which is 12,500 bytes long, and WMAN, which is 7,800 bytes long, and all explained in the documentation.

German

If you boot QD from the supplied medium you will find that the program will appear in German. Have no fear because, for all who do not know German, there is an English version included on the disc. Load the boot and edit

line 30 to read EXEC flp1_QD_ENGLISH. Then type run and press ENTER and the English version will load. The disc is not copy-protected,

Information:

QD
Price £26.45
Jochen Merz Software
Im Stillen Winkel 12
4100 Duisburg 11
West Germany

so you can easily make a back-up of it with a customised boot for your use.

There is a configuration program which is simple to use to configure QD but do your configuring on a back-up copy — never on the original.

Before you run the program read the documentation thoroughly. It is friendly and has much to offer. QD needs experiment, like any other program, to get a feel of the controls and to see how the options

work and what they do when executed; as far as I am concerned it is the only way to get the best from software.

QD would be useful for those who program in machine code, Pascal, C, BCPL, Fortran or any of the other languages which require an editor. It can multi-task with other programs, so you can have your assembler-linker and so on laying in memory with QD, if your QL memory permits, and switch between them with CTRL + C. There is no reason why you could not have two or more QDs in memory at the same time.

Basic edit

You can also edit Basic programs with it. I have tried it and found no problems and some of the programs I loaded were very long. When finding and replacing, QD took it all in its stride and, despite the file length, it did it in the blink of an

eye. Apart from using it in a programming environment it could also be used for simple things like keeping notes, records of appointments, recipes or anything involving the manipulation of text.

No Quill

One thing it did not seem to like was Quill documents. No matter how short the file length, it would load about 160 characters of the file and no more. I do not regard this as a bad point of the program, even though there are text editors which will handle documents which have been produced by Quill. QD is neat and efficient to look at. It is made easy to use and friendly, as it is pull-down, menu-driven and I like programs which use those types of menu.

At the top of the screen are small windows with words such as FILES, COMMANDS, BLOCKS, INFO, STATUS in them. They are the categories with which the menus deal. There is also a group of four boxes at the top left of the screen. They relate to a loadable help page, re-sizing the QD window, moving it, and leaving QD temporarily to enable a return to it with CTRL + C.

The text cursor is moved round the editing window with the normal cursor keys, one line up or down, character by character, back or forth along a text line. Pressing ALT, SHIFT, CTRL or a combination of them with the cursor keys allows you to do such things as delete text, character by character or line by line, move to the start or end of text, move along a line word by word, scroll the text page by page, or pan the text left or right leaving the cursor in its current position. There are 22 cursor-orientated commands regarding movement and editing.

By the side of the text window there is a narrow window, the height of the text window, just wide enough to accept the cursor. It is used for fast selection of an area of text for viewing or editing.

Selecting menus is simple. Move the cursor out of the text area to a category and press ENTER and the appropriate menu will be pulled down. Options are chosen from the

menu with the arrow or the first letter of the required option.

Once a menu has been pulled down, further menus cannot be pulled down until the menu is pulled up again and this also applies to the execution of an option selected from a menu. Menus can also be selected using the function keys which are displayed by the side of the appropriate windows.

INFO is not really a menu but an information sheet which tells you current line, column and free memory.

STATUS allows you to change parameters, such as tabulation intervals, used by QD and set insert mode, confirmation request and file backup.

When the insert mode is on text can be inserted and the original text to the right of the cursor is moved to make room for it.

When the insert mode is off the text underneath the cursor is over-written by what you type-in. If you want to insert any text you can make room for it by putting spaces in using SHIFT + SPACE.

BLOCKS allows you to mark blocks of text to copy, delete or move them. Marked blocks can have strings replaced in them and be written to a device such as disc or Microdrive, or to a printer by specifying ser1 or ser2 instead of a device and filename.

COMMANDS will allow you to find/replace strings upwards or downwards throughout a current text file, move to specific lines, delete or insert control codes or quit QD. If you use the

One thing it does not seem to like is Quill documents . . . I do not regard this as a bad point. QD is neat, efficient and easy to use.

quit option all the text is lost and you cannot re-enter QD until it has been loaded again.

FILES, as the name suggests, is all to do with file handling. Files can be loaded, saved and sent to a printer. If you wish, new fonts can be loaded so long as their file names are appended with _FNT. Special fonts, too, can



be loaded which use CHR\$(0)-(31). High-res fonts such as those of *Page Designer*, or those supplied with *Qwriter II* are not suitable for use with QD.

QD allows you to enter text up to 160 characters per line and scrolls the text off to the left if you type-in a line with more characters than it can display in its text window, which is

Shirley Jane Peters tries out a new text editor, QD, to date the only one on the market to run under the QJump pointer environment.

Files which have been made by another editor can be loaded into QD. If they contain control codes such as CHR\$(8) or CHR\$(13) those codes can be removed. Also it is possible to insert control codes into your text with the option Put Character.

The insert text mode in this menu I found really useful because it means that I can merge a file from disc or cartidge with the current file in QD and put it anywhere I want to do within the text at the current cursor position. When I program in machine code I usually

standard practice with text editors.

It is a fast program because it does no garbage collection and memory is allocated only as it is needed. In this respect it is ideal for the programmer who needs to use an editor. Speed was one of the considerations of the author, and as he said in the manual, "Programmers want fast things; they have no minutes to wait," and he has certainly achieved it.

No bugs

I understand that there are many text editors on the market but which one you use depends on if it does what you want. I found QD to be more than adequate and, with the amount of memory I have on my QL, there were no problems. No bugs reared their heads. The program offered me options you find on many text editors, plus a few more.

It is easy to use, pleasing to the eye, well error-trapped and fast in the way it carries-out its tasks. What I have seen from this company makes me think I would like to see more of its products, I like its software. If you need a text editor and can make room in your disc box for this one. I do not think you would be sorry.

INFORMATION:

Product: FlashBack Special Edition.
Price: £40 (£15 upgrade charge for users of FlashBack 1).
Supplier: Sector Software, 39 Wray Crescent, Ulmes Walton, Leyland, Lancs PR5 3NA.
Tel: 0772 454328.

FLASHBACK S.E.

Bryan Davies with a user report on the new V2 update of this popular database manager from Sector Software.

A reviewer may see a program in a different light from some users, partly because the reviewer tends to change to the latest program versions as soon as they are available — often before they are released to the public — and partly because the period during which a program is tested can be short. In the case of *FlashBack*, I have been using it since it was introduced and have had the version reviewed for several months; any bad features are likely to have shown up by now. As the new features of the current version have by now been accepted by me as normal it also means that it is not difficult to overlook the fact that readers will be unaware of them.

While *FlashBack* has been enhanced, many existing users will be more concerned with what is in the Special Edition package. There are three modules supplied with *FlashBack* but usable independently of it. One is a Report Generator, another a Forms Designer and the third a Printer-Driver Customiser. The modules will be dealt with in the concluding part of this review in a later issue.

Original

The original *FlashBack* is still available at £25 and may be more suitable for users with QLs which do not have memory expansion units, because it uses significantly less memory than the new version. Existing users can upgrade to the full new package by sending £15 and proof of original purchase to Sector Software.

FlashBack never set out to replace *Archive* but one of the intentions was to provide a database program which was easier to use than *Archive* for typical applications. The program is very successful in that; my test was to transfer a name and address file — 300 records — and a small German dictionary — 1,900 records — from run-time *Archive* to *FlashBack* and the results were so good that I no longer use *Archive* on a regular basis.

Apart from greater ease of use the big advantages are speed of operation and flexibility of data manipulation. The ability to make notes, or even sizeable documents, and deposit them later into a word processing program gives *FlashBack* another distinct edge on *Archive*. Perhaps the most impressive feature, especially for someone who habitually loads the QL until there is little memory left, is the consistent high speed of both cursor movement and string searches. The cur-

sor is not only fast but also very easy to follow. Regardless of file size, typical searches each take roughly the same time — a second or so.

The two files mentioned have no data in common; the one is strictly names and addresses in one field with telephone numbers and odd information in another, whereas the second one has German words in one field and the English equivalents in the other. As both are sometimes required concurrently there was a problem when using *Archive* — any swapping routine took far too long. Not so with *FlashBack*; the German dictionary was put as Fields 1 and 2, the names file as Fields 3 and 4, and the two files merged into one.

Merging *Archive* files, or just doing a one-for-one conversion for use in *FlashBack*, is straightforward. During operation it is simple to Group the records so that the one set can be excluded during searches but the speed is such that this feature is rarely necessary; with the database file size at 140KB, operation is as slick as when the size is only 40KB.

One disadvantage needs to be mentioned. The total space taken by program and database is considerable — in the example quoted, roughly 140KB for the file and 130KB for the program. The combined database file size is about 20KB less than it was in *Archive* and the 1:1 ratio improves as the database grows but it may be difficult to justify the new version of the program if only very small database files are used.

The program space can be reduced by making the window used smaller, not taking the option to refresh the screens, reducing the number of references permitted per record, and by limiting the number of records which can be added during one session; more than 30KB can be saved in that way. Although it would in a sense be a retrograde step, it might be worth having the facility to keep the data on disc instead of in memory, especially now that it looks as if hard discs will be available to give much faster access than floppies.

New features

The basic commands are unaltered and existing files can be loaded, so there is no difficulty upgrading from version 1. A useful facility if there is a string of records beginning with the same word is the ability to go to the first of them by pressing F2 when the cursor is on that word in another record. If you are on the last of, say, 50

records each beginning with Smith, place the cursor somewhere on the word Smith in the current record, then press F2 to go to the first of the 50. The word does not have to be the first in the current record but the search action always goes to the record with that word as the first one.

An instruction database file is supplied and it can be incorporated into any user database to give on-line help. The *FlashBack* code is now re-entrant, which allows more than one copy of the program to be run at any time, using the same program code. When the commands are listed, by pressing F3 twice, the first on the list is "Another F1BK — Ctrl A".

Default

Naturally, different keying will be required for the second copy and you are first asked what that new keying is to be. The default database is then loaded or you can specify another. The memory required by this second copy is roughly 50KB — for index space and so on — plus the database. Further copies can be loaded from either of the two and copies can be called up one on top of the other. Copies can be quit individually.

The illustration shows at the top the Q_Switch status indicators, (background), this article being written in text⁸⁷, (bottom) the first *FlashBack* main window, (middle right) the first *FlashBack* secondary window, and (top right) the second *FlashBack* main window showing another record from the same file as is being used by the first copy.

Some new keying has been provided. The next record can now be obtained either by Ctrl+N or by Alt+Down and the previous one by Ctrl+B or Alt+Up. Instead of having to re-start a Search operation from scratch after each instance of a string is found, the next instance can be found by using Alt+Shift+Down. A Replace function has been added to the Search; when any search string is specified you are prompted also for the replace string but pressing Enter without inserting a replacement string will initiate the search on its own.

Search/Replace operations can be from either the current record on, or from the start of the file, depending whether initiated by Enter or Shift+Enter. The Index command is comparable with the *Archive* Order command. The Field on which indexing is to be based can be selected and there are three possible sorting orders; the default order is all alpha upper-case first followed by all

alpha lower-case, with the alternatives being upper- and lower-case for each letter and "as they come."

This is best illustrated by example — i) ABCabc, ii) AaBbCc and iii) aBcDef. For a dictionary, option iii is appropriate. Numbers can be treated in two ways. The default is to treat them as "text", as in FlashBack 1, so that 99 will be placed later in the order than 100, because the first character, 9, is later in the code table than 1.

When the Index command is selected you can choose to have numbers treated in their normal numeric order — i.e., 99 before 100 — instead. Successive Group actions can be taken, down to the point where only one record is current. In a names file containing both personal and business contracts it is possible to select business names by type, then carry-out further selection among them by specifying a particular product or brand. The following record is a fictitious example:

Field 3: Autoway Motors, 11 Pickington Road, Beltringham.

Field 4: Tel: 01-999 1212 (cars;Audi/VW-Rod Holliday,sales mgr.)

By grouping on "cars;", all such records would be made current, to the exclusion of any not involving car dealers. A further grouping, using "Audi" as the string, would eliminate any records of dealers not handling that brand. If only those dealers based in the London area are required, a further grouping could be made on either "London" or "(01)". To carry this to the limit, grouping on "Rod" would be likely to reduce the number of current records to one. Printing can be based on such groupings, only the currently-selected records going to the printer.

As you left it

A feature which always impresses me, small though it may be, is that each time you re-load FlashBack the cursor is where you left it at the end of the previous session; further, the last strings used with commands are there when those commands are called up again. The size and location of the window are as set the last time the Size command (Shift+F4) was used. A very useful improvement over the original is that the current string does not have to be deleted to enter a new one, as the latter causes the former to be removed automatically.

The status field is different now, having three indicators — C, I and G. The "C" appears in white when a change has been made to the file, the "I" appears likewise if the file is indexed, and the "G" also if the file is grouped. A point which will please Q-Switch users is the change of keying from Shift+Caps Lock to Ctrl+Caps Lock to give case-dependence when using the Group or Search commands.

The contents of database files can be written to disc in sections. If there are

several blocks of data in a file which need to be written-out separately, the Group command is used to select the blocks one at a time and the Write command is then followed by Shift+Enter, rather than Enter, as you would write out the whole file.

Two other new commands — Xclude and INclude — allow records to be treated separately from the Group criteria. A record which would meet the grouping criteria can be previously marked by Xclude and it will then not appear in the group; on the other hand, a record which does not meet the grouping criteria can be marked by INclude and it will then appear in the group.

Grouping can be used for deleting large numbers of records; the records not to be deleted can be grouped and written-out to disc/cartridge, then read back in. The unwanted records will effectively have been deleted by the selective writing.

A Quit command has been added. When it is used the program, indices and database file are removed. Although a check of free memory before and after may show no change, the space has been vacated, as can be proved by EXEC-ing

figuration process for adding records, typically only a few KB. By using the Import process Quill files in a similar format to Abacus files can be handled at any size.

There is considerable choice in the layout of the fields for the __dba file — Archive or Abacus fields can be merged during the conversion or several fields can be put on to the same line. If a one-to-one conversion of fields is sufficient all that is necessary once the program has started is to press the ESC key for conversion to be made automatically.

There is a concept available which is not found in Archive, referred to as "sub-records". The example given is of a gramophone record catalogue, where each album may contain tracks by several artists/groups. In Archive, it is necessary to give each track, title and artist separate fields, to avoid problems when searching for particular titles or artists. FlashBack allows each occurrence of either title or artist on an album to be treated as a sub-record — i.e., track titles may be sub-records 9.1, 9.2 and so on. The Import routine offers the option to amalgamate

REVIEW: FlashBack 2

PRODUCT : FlashBack 2
PRICE : £??
SUPPLIER: Sector Software
39 Wray Crescent
Ulmes Walton
Leyland
Lancs. PR5 3NR.
Tel. (0772) 454328

DCreative Codeworks P.O. Box 1095 Birmin
Tel. (021)-426-5199 (GL;Simon Goodwin,
Gus Chandler-Speedscreen,Quickfax??)

DSector: Software 39 Wray
Batty-Taskmaster,SpellBo
und,FlashBack,Page
Designer,OverDrive,OmniD

04 PHONE 004972
00075
I - flp2_R_dba

04 PHONE 004922
00389 00389
C I - flp2_R_dba

A reviewer can see a program in quite a different light to some users, partly because the reviewer tends to ch

DSector: Software 39 Wray Crescent Ulmes Walton Leyland Lancs. PR5 3NR. Tel. (0
DSector: Software 39 Wray Crescent Ulmes Walton Leyland Lancs. PR5 3NR.
Tel. (0772)-454328 (GL;David Batty-Taskmaster,SpellBound,FlashBack,Page
Designer,OverDrive,OmniDump,Tough Typist)

03 PHONE 004922

00389 00389
C I - flp2_R_dba

the program again. You can, therefore, remove FlashBack temporarily if memory is needed for other jobs, without the need to run the FlashBack boot to load it again later. There may be a fall of about 12KB in free memory when the program is re-run.

The flexible design of FlashBack permits files from Quill, Archive and Abacus to be Imported in __dba file format, after they have been converted by a straightforward Import routine — a separate program. This feature has been available from the start but was previously suggested for use with Archive only. Quill files can be Merged, provided they are in text-only format — usually with the extension __lis — but the file cannot be larger than the memory space allocated in the con-

sub-records after records have been converted. It is then possible to search the one field — number 9 in this example — for any track title, rather than having to search several fields.

Users have asked for additional features, such as a report generator. This, and a form generator and a printer-driver customiser, are supplied as stand-alone modules which can be used with FlashBack. The general layout and specification of the new modules looks good. The command lines are clear and largely self-explanatory to those brought up on Quill and Archive. As with FlashBack, each of use is a prime consideration. These modules will be dealt with in the concluding part of this review.



SUPER BASIC

In the fourth part of our calculator/base conversion project, Mike Lloyd wraps up the main program.

Keen followers of the Super-Basic QL Calculator project will by now have the screen format and input routines for the program entered into their computers. By adding this month's listings the project will be almost complete and will certainly be usable, with only the luxury extras offered by the menu options outstanding.

The listings accompanying this article perform all the arithmetic undertaken by the QL Calculator and they also format and print the output. Figure one shows the now familiar hierarchy formed from the relationships between the modules. The definitions on the lowest level support the modules on the next level up, which in turn are called from the root procedure at the top of the hierarchical tree.

Tree analogy

The tree analogy for this type of structure is so widely-accepted that it seems churlish to point out that roots are normally found at the bottom of trees, not at the top. Root has been used to denote the parent of any hierarchical data set

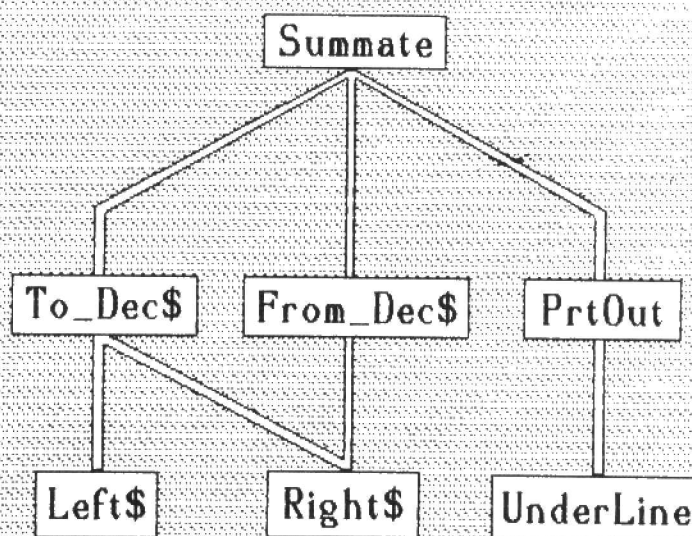


Figure 1: The Hierarchical Relationships of the Modules

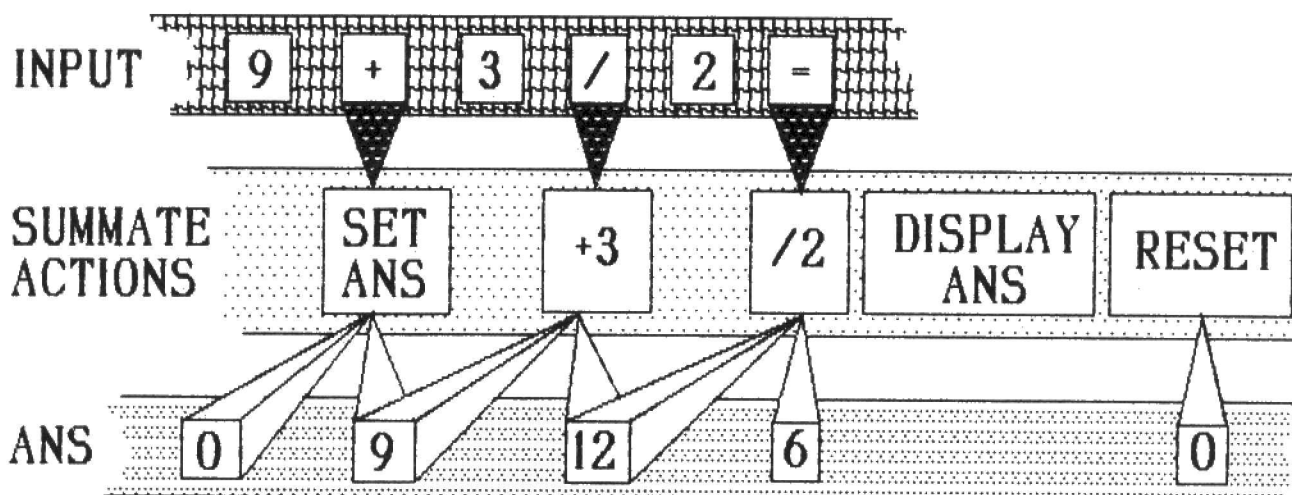


Figure 2: How the Summate Procedure Responds to the Input Stream

almost since the beginnings of computer science. It has established such credence that in the Unix multi-user operating system the person with the most privileges and ownership of the ultimate file directory is universally known as root. It is too late to insist on changing the analogy for one more faithful to nature.

Parented

Even with such a compact hierarchy as that parented by the *Summate* procedure all of the advantages of structured programming are present. Needless repetition of code is avoided by putting frequently-used algorithms into utility procedures called from various other definitions. This concentration of a single task into a single place in the program code also means that any changes to that algorithm can be introduced by amending the internal workings of a single segment of code rather than by making alterations throughout the program.

Program readability is enhanced further by dividing the code into coherent segments and by referring to any complex subordinate process by its user-provided name.

Turning away from the inter-relationships between the definitions and towards their internal workings, the *Summate* procedure in listing 3.1 is devoted to executing each step, or transaction, in a chain of calculations. Contrast this with the role of the input routine listed in the previous instalment and you will see that the program is switching constantly between receiving information and acting on it.

Such alternating reactive and pro-active phases are very common in real life. Chess players alternate between making a move and waiting for an opponent's response. Tennis players are either receiving the ball or returning it to an opponent. Armies are either attacking or defending. People either influence events or are influenced by them. The distinction between acting and reacting becomes less well-defined with each example but the essential principle remains true.

The key to understanding the workings of the code is in the use to which variables are put. Characters recognised as valid digits are placed in a variable called *Num\$* by the input routine listed earlier in the series. *Num\$* can be in any base from binary to hexadecimal and so its decimal equivalent is calculated and placed in the variable *DecNum* before any arithmetic takes place.

For each transaction two operators normally are involved. The one which triggered the call to *Summate* is referred to as *Key* while its predecessor is known as *Op*. The numeric variable *Ans* — short for answer — holds the running total for each calculation until eventually it is displayed as the final result when the equals key is pressed. Figure two shows why it is always the previous operator which is used to update the running total.

QL Calculator recognises four classes of arithmetic transaction — the first in a sequence, "running" transactions, sub-totals and the full total, which is always the last transaction in a sequence.

For the first transaction, when the value of *Op* is 18 — i.e., it represents a blank — the running total is made equal to the first value to be entered. Subsequent "running" transactions cause *Ans* to be modified but it is not displayed until the equals sign is entered, when it is shown in the input area in the input base. If the next keypress is an arithmetic operator the sum continues. Pressing the equals key a second time forces the total to be converted to the output base and the variables zeroised in readiness for the next calculation.

Base conversion algorithms were explained in the introductory article for the project. The functions appear again here in an adulterated form to cope with the special requirements of the program and with an important enhancement — they now handle negative values. The following explanations cover only the modifications to the general-purpose functions.

The code at listing 3.2 turns any number into its decimal equivalent. The standard algorithm is altered only to dispose of floating point decimal numbers and to convert negative integers. It might seem pointless to "convert" a number which is

Adding this month's listings will make the project almost complete and certainly usable, with only the luxurious items outstanding.

already decimal but the statement at line 3215 is not the simple *IF base = 10 THEN RETURN Number \$* which might have been expected.

Instead, it allows decimal integers to go through the conversion process. This has the satisfactory side-effect of allowing decimal integers with up to eight digits to be displayed without resorting to the annoying "E" notation.

Valid digits

The conversion routine expects a string consisting only of valid digits but the *Num\$* string might also have leading zeros and a unary minus. The *INSTR* function on line 3220 is used to detect and locate a unary minus. If one is found the temporary string *Value\$* is sliced from *Num\$* to the right of the minus. Alternatively, *Value\$* is defined by the *Left\$* function. The conversion algorithm can now set to work; the result is multiplied by minus one if necessary and returned to the calling statement.

Conversion in the other direction is

more complex. The decimal number is held in the *Number* variable and its absolute — positive — value is copied to the temporary variable *n*. A string of blank characters, *X\$*, is prepared to receive the final result.

FromDec\$ is designed to receive integer or floating point numbers even though the conversion algorithm copes only with integers. The first special case to be handled is that of tiny fractions which the QL would represent in scientific format; they are all reduced to equal zero.

The second special case handles numbers which are too large to be displayed on the program portion of the screen. Three stages of overflow exist. A number might have too many digits only because it is represented in a small base, such as binary. Alternatively, it might be so large that its scientific notation exceeds 10 digits. In other words it is greater than one followed by 99 zeros. Finally, it might be so huge that the QL crashes.

No billions

In the first circumstance an "overflow" flag is set and a decimal number is returned. If a decimal number exceeds 10 digits it is trimmed to fit the display area by splitting the number at the "E" and removing as many digits as necessary from the left part before joining the number together again.

Accuracy is reduced only slightly because such huge floating point numbers are approximations anyway. Sadly, nothing can be done about super-large numbers and the program will inevitably crash if one occurs. Calculations involving billions of billions will have to be undertaken elsewhere.

By the end of the long *IF* statement at Line 3344 the routine has disposed of all extreme numbers, leaving the following three categories:

- ★ Values which can be displayed in eight or fewer digits in a non-decimal output base.
- ★ Values which would exceed eight digits if they were converted into the output base.
- ★ Values which are to remain in decimal.

The first category is dealt with by the conversion algorithm detailed in the first instalment of the series. For both of the remaining categories the output is a decimal number, with the overflow flag being set if the output is a failed conversion to a non-decimal base.

The arithmetic is now all over, so non-mathematicians can address the more practical problems of tailoring the output to suit the program constraints. Listing 3.4 copes with all output to the screen and the printer. The routine begins by establishing a blank string of 14 characters, into which all output values will be concatenated, and by scrolling the screen display area to make room for the new display line. Output can be a number or a message, the

latter being distinguished by an asterisk as the first character.

PrtOut appends to each non-blank output string the base identifier — *h* for hexadecimal, *b* for binary and so on — and the appropriate operator. Overflows are indicated by the presence of a question-mark instead of the base identifier. For printer output, division is represented by a slash because Epson printers do not include the conventional division sign in their character sets.

Listings 3.5 and 3.6 are general-purpose string functions which remove and replace leading blanks. *Right\$* adds blanks to the beginning of any text to pad the string length to exactly 10 characters. *Left\$* removes leading spaces for the benefit of the *To_Dec\$* function but it could be used whenever left justification was required.

The final procedure copes with underlining for output to screen and printer. The underlining algorithm states that a subtotal is preceded by an underline and a final total is also underlined. Underlining therefore takes place whenever the equals sign is pressed except when the *Op* variable is set to 18, indicating that the equals sign is not being used to complete a sum.

Once the accompanying listings have been added to the program, the QL Calculator will work in floating point mode. By changing some of the values in the *Init_Var* procedure there can be experiments with different input and output bases. The values chosen must accord with the advice given in figure three.

Menu options to automate this process and to accomplish much more besides will be published next month to bring the project to its conclusion.

INPUT		OUTPUT	
IN_BASE	IN_VAL	OUT_BASE	OUT_VAL
10	1	DECIMAL	10 1
16	2	HEXIDEcimal	16 2
2	3	BINARY	2 3
8	4	OCTAL	8 4

IMPORTANT: THE VARIABLE `INTONLY` MUST BE SET EQUAL TO ZERO ONLY IF BOTH THE INPUT AND THE OUTPUT IS DECIMAL. SET `INTONLY` TO 1 IF A BASE OTHER THAN DECIMAL IS BEING USED.

FIGURE 3

Listing 3.2

```

3200 DEFine FuNction To_Dec$ (Base, Number$)
3205 LOCAL y, z, Value$(10), digit$(16), Break
3210 digit$ = "0123456789ABCDEF"
3215 IF NOT IntOnly: RETURN Number$
3220 Break = "-" INSTR Number$
3225 IF Break
3230 Value$ = Number$(Break +1 TO)
3235 ELSE
3240 Value$ = Left$ (Number$)
3245 END IF
3250 z = 0
3255 FOR y = 0 TO LEN (Value$) -1
3260 z = z +(Base ^y) *(Value$ (LEN (Value$) -y)
INSTR digit$ -1)
3265 END FOR y
3270 IF Break: z = z * -1
3275 RETURN Right$ (z)
3280 END DEFine To_Dec$

```

Listing 3.1

```

3100 DEFine PROCedure Summate
3102 LOCAL DecNum
3104 DecNum = To_Dec$(In_Base, Num$)
3106 SELECT ON op
3108 = 18: Ans = DecNum
3110 = 20: Ans = Ans - DecNum
3112 = 21: Ans = Ans + DecNum
3114 = 23: Ans = Ans * DecNum
3116 = 25
3118 IF DecNum <> 0
3120 Ans = Ans / DecNum
3122 ELSE
3124 Warning: PrtOut "*" Div by 0"
3126 END IF
3128 END SELECT
3130 PrtOut From_Dec$ (Out_Base, DecNum)
3132 Num$ = B$
3134 SELECT ON Key = 19 TO 24: op = Key + 1
3136 IF Key = 26
3138 SELECT ON op
3140 = 20 TO 25: op = 26
3142 Num$ = From_Dec$ (In_Base, Ans)
3144 IF Overflow
3146 PrtOut Num$: op = 18: Num$ = B$
3148 END IF
3150 = 26: op = 18: Ans = 0
3152 END SELECT
3154 END IF
3156 END DEFine Summate

```

Listing 3.3

```

3300 DEFine FuNction From_Dec$ (Base, Number)
3304 LOCAL ex, n, x$(10), digit$(16), Temp$(16)
3308 digit$ = "0123456789ABCDEF"
3312 n = ABS(Number): x$ = B$
3316 IF n < 1E-6: RETURN Right$ ("0")
3320 IF LEN(Number) > 10
3324 Overflow = (Base <> 10)
3328 Temp$ = Number: Break = "E" INSTR Temp$
3332 x$ = Temp$: Temp$ = Temp$ (Break TO)
3336 x$ (12 - " " INSTR Temp$ TO) = Temp$
3340 RETURN x$
3344 END IF
3348 IF Base <> 10 AND n < Base^8
3352 IF Number<0: x$(9-INT(LN(n)/LN(Base))) = "-"
3356 FOR ex = INT (LN(n)/LN(Base)) TO 0 STEP -1
3360 x$(10 -ex) = digit$(1 +INT(n /Base ^ex))
3364 n = (n -(Base ^ex) *INT(n /Base ^ex))
3368 END FOR ex
3372 RETURN x$
3376 END IF
3380 Overflow = (Base <> 10)
3384 RETURN Right$ (Number)
3388 END DEFine From_Dec$

```


Listing 3.4

```

3400 DEFine PROCedure PrtOut      (Text$)
3404 LOCAL Temp$(14)
3408 Temp$ = FILL$(" ", 14)
3412 AT 8, 0: SCROLL -10, 1: AT 7, 0
3416 IF Text$(1) = "*"
3420   Hue 2: Temp$ = Text$
3424 ELSE
3428   Hue 4: UnderLine
3432   IF Overflow
3436     Temp$(11) = "?"
3440   ELSE
3444     IF Text$ <> B$
3448       Temp$ (11) = BaseTag$ (Out_Val)
3452     END IF
3456   END IF
3460   Temp$ (13) = Valid$ (op)
3464   Temp$ (1 TO LEN (Text$)) = Text$
3468 END IF
3472 CLS 3: PRINT Temp$(1 TO 13): AT 8, 0: UNDER 0
3476 IF PrtOn
3480   IF Temp$(13) = CHR$(187): Temp$(13) = "/"
3484   PRINT#5; Temp$ (1 TO 13)
3488 END IF
3492 END DEFine

```

Listing 3.5

```

3500 DEFine FuNction  Right$      (Text$)
3505 IF LEN (Text$) > 9: RETURN Text$
3510 RETURN FILL$(" ", 10 -LEN (Text$)) & Text$
3515 END DEFine Right$

```

Listing 3.6

```

3600 DEFine FuNction  Left$       (Text$)
3605 LOCAL x, Temp$(10)
3610 Temp$ = Text$
3615 FOR x = 1 TO 10
3620   IF Temp$(x) <> " ": EXIT x
3625 END FOR x
3630 RETURN Temp$ (x TO 10)
3635 END DEFine Left$

```

Listing 3.7

```

3700 DEFine PROCedure UnderLine
3705 LOCAL x$
3710 UNDER (Key = 26 AND op <> 18)
3715 IF PrtOn
3720   x$ = CHR$(27) & "-" & (Key = 26 AND op <> 18)
3725 PRINT#5; x$;
3730 END IF
3735 END DEFine

```


TRANSFERS without tears

I There are many ways to move files between a QL and a PC. You pay your money and you take your choice. Bryan Davies reassesses the file transfer programs available to date.

There were several developments in the area of file transfer during 1988 and 1989 started with some more. You can now move a wide range of file types between a home-based QL and an office PC — and in the reverse direction. With certain file types you can avoid the chore of having to re-insert printing codes. The ultimate in theory at least is to be able to use the same files, on the same discs, on both types of computer, and that also is now possible.

What is not possible at the moment is to transfer spreadsheet files complete with formulae; the data can be transferred but the formulae are lost. Neither can programs be taken from one computer type and run on the other but, if you can write in SuperBasic and have compilers for the C language, you can do the writing on the QL and the running on both QL and PC.

All the programs mentioned run on the QL — you do not have to buy anything extra for the PC nor do you have to do anything special on it; the PC and the programs you run on it are used in the normal way to load and save files. There is no need for inter-connecting cables; you take discs from PC or QL and insert them in the other system.

In the case of the emulator, you do not even need a PC. There have to be reservations, of course, but they are not due primarily to the transfer programs. The disc modes and formats used on PCs are somewhat different from those used on the QL and, although allowances are made for this in the programs, they can do nothing about the fact that QL drives are not equipped to handle high-density discs.

Much as you might fancy the idea, you cannot increase your QL disc capacity to 1.2MB or 1.44MB; a suitable interface could be designed but existing drives are almost certainly unsuitable for working at the higher densities and new drives would not be cheap — typically £150-£200 for a single 3.5in. drive, without interface or power supply, on a PC.

The basic difference between the PC 40-track and QL 80-track 5.25in. drives is dealt with by the software, but you may have to provide some information to enable the programs to make the necessary changes.

The two dominant operating systems on micros are CP/M, as used on the Amstrad PCW series and Commodore 64, and PC-DOS/MS-DOS for IBM PC/XT/AT models and their clones. In terms of numbers of users, there may be more call for CP/M-related programs than for MS-DOS ones and CP/M is catered for by the programs mentioned. MS-DOS is much more the current interest than CP/M except where games are concerned and it is assumed that readers are concerned primarily with transfers to and from a PC in the IBM or compatible sense.

Inevitably, the emphasis is on the transfer of text files. You can transfer spreadsheet or database files but the programs available do not offer facilities for making them easily usable afterwards. With the formulae or the layout lost you might well think the effort of re-creating them is not worthwhile; it is better to start from scratch and create new files in the other machine.

That does not apply to an emulation program. As the files are unchanged in the transfer process you can continue working on them as normal, whether they be word processing, spreadsheet or database files. Although not considered in the context of this article, the

Import program supplied with *Flashback* provides a means of taking Quill, Abacus and Archive files and putting them in a format suitable for use in FlashBack, without further attention; no provision is made for transfer in the reverse direction.

What are the reasons for buying an emulator? Emulation programs cost about the same as other major programs and, no doubt, many copies will be sold to users who buy out of interest rather than to meet a specific need. Perhaps the user is interested in seeing what programs for

other machines are like, or how the machines operate. Buying an emulator is cheaper than buying the other machine and gives the user a chance in part to evaluate something he may be considering buying.

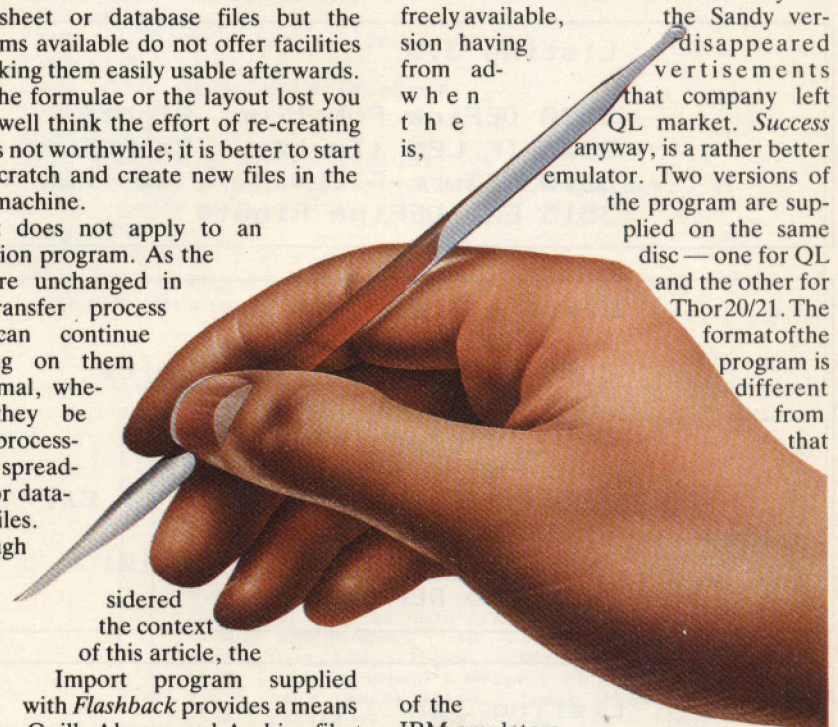
It is certain that some QL users will have had a CP/M machine and will have programs used on that; some of the programs may still have no direct counterpart on the QL and a CP/M emulator provides the facility to run them.

There are also PC users among the QL fraternity and they will already have a version of MS-DOS, plus programs such as *WordStar* and *SuperCalc*. Some of those users will work on a PC in their offices and may want to continue some jobs when they get home. Others may feel that learning to use a PC will be an inevitable part of business life in future, so why not look at it now, in the relative peace and quiet of the home QL atmosphere?

Swapping files by means of a conversion utility enables work to be taken to and from the office but the time taken by the conversion process may be unacceptable if extensive editing is required to get the transferred file to print-out with the same typestyle enhancements.

Success

This CP/M emulator is now the only one freely available, the Sandy version having disappeared from advertisements when that company left the QL market. *Success* anyway, is a rather better emulator. Two versions of the program are supplied on the same disc — one for QL and the other for Thor20/21. The format of the program is different from that



of the IBM emulators, in that you do not run a copy of the CP/M operating system after it, because version 2.2 of that OS is emulated in *Success*.

All the features of CP/M are there. All you need is *Success* plus CP/M versions of your favourite programs. You can also write and run Z-80 assembler programs. It is simple to get started if your CP/M discs are in a format *Success* accepts immediately. Microdrives and RAM discs are still usable devices

but CP/M programs must be on disc.

If the disc format is not accepted initially you have to go through a conversion process, which may be fairly straightforward for a hacker but may not be so with some disc formats for a non-technical user. At least one disc drive and 640KB of memory are required to run Success. Speed of operation is quoted as equivalent to a 2MHz Z-80 processor; that is similar to that of many CP/M machines, so is fast compared to the MS/DOS emulators. Qdos and CP/M jobs can be multi-tasked. Apart from the large commercial library of CP/M programs there is a vast range of public domain software available.

The Solution

The attractions of an MS/DOS emulator are similar to that for CP/M emulation — a large range of commercial and public domain software is available and the user may have several programs as a result of previous involvement with those machines. PC-DOS can be taken as being the same as MS-DOS so far as the programs which run with them are concerned; PC-DOS is used only on IBM machines.

MS-DOS offers little joy as an operating system. It rarely rates favourable comment from journalists; compared to Qdos it is a pain. The only thing which really counts for it is that it is firmly associated with IBM and the PC and there are an estimated 15-20 million computers round the world using it.

The Solution can be expected to work with most versions of MS-DOS and has been tested with various stages of versions 2, 3 and 4. The emulation approach is, as with Success, to use software to simulate the PC hardware but the difference is that the operating system is not emulated but is run separately in the same form as it is on a PC — the emulation program is set running and it then runs MS-DOS from a separate disc.

An interesting aspect of this non-hardware approach to emulation is that there should be no argument about whether or not the BIOS, which is on ROM in PCs, is a copy of the original, because it is written in 68000 code and, therefore, looks unlike 8086 code. You need at least one disc drive and a minimum of about 384KB of memory. As PC programs tend to be much larger than their QL equivalents you really need to have a Trump Card fitted to be able to run some of them.

Microdrives and RAM discs are not usable. Speed of operation is quoted as being typically 10 percent of that of a standard PC-XT, which is rather slow. The perceived speed varies considerably with the program and type of job. There is a noticeable lag in characters appearing on the screen when using a word processing program but the usual input buffer allows you to type well ahead of the display without losing characters; response to command keys is good.

When word processing, most of the processor time is spent reading in keyboard input or waiting for it and processor speed is not the most important factor; under those circumstances effective speed may be as high as 90 percent of an XT.

Coming from a fast AT with fast hard disc to the emulator, the slow-down is marked but that is true when the QL is used in its normal Qdos mode. On the other hand, if you have been used to a basic IBM PC with only floppy drives, you will not feel the change so much. For those who habitually use the basic Quill on a QL which is running other jobs at the same time there may be little difference.

For typical files, a slower speed is acceptable but writing a long book could become a chore. As loading and saving are a major part of word processing operations, the times taken for them are of similar importance to processor speed and 360KB floppy drives make the PC a slow machine. The advent of hard disc im-

will prove operating speed with the emulator: it is hoped that partitioning of the hard disc will be possible between Qdos and MS-DOS and the latter already allows partitions for other operating systems such as Unix.

The big thing is that you take not only the program discs but the data ones also directly from a PC and put them into the QL drives and continue working on the same files, taking them back to the PC after editing. The emulation even has some advantages over the PC; the DOS area available to the user rises from 640KB to 667KB — maximum, on an 896KB system — and the disc drives can be 80-track if you have them and give 720KB storage space on the 5.25in. disc.

You can multi-task Qdos and MS-DOS if your system has sufficient memory. At the moment I am typing into *text87*, running under Qdos and watching Quill text appearing on the screen simultaneously under MS-DOS; there is no doubt the QL is multi-tasking, because two portions of text are being written to the screen at the same time. Disc drives can be allocated separately to Qdos and MS-DOS; the latter can be booted from either drive, the boot drive being regarded by it as A:.

It is recommended that you use a speed-up program such as *Lightning* to improve the rate at which characters are displayed on the screen and also the speed of processing. Appreciably faster running is obtained on the Thor XVI and the Atari QL Emulator — claimed

to be 2.5 times as fast as on the QL.

Compatibility is a familiar word in the PC world and *The Solution* creates no significant problem. It will run almost all the well-known programs — *Lotus 1-2-3*, *WordStar*, *WordPerfect*, and *Word*. Those it cannot run may well not run on PC clones either. There are practical restrictions, which apply equally to PCs; graphics files created using an EGA screen cannot be displayed, because the emulation is of a CGA display. Programs requiring keyboard input as found in *Flight Simulator* cannot work, although the demonstration mode of FS works.

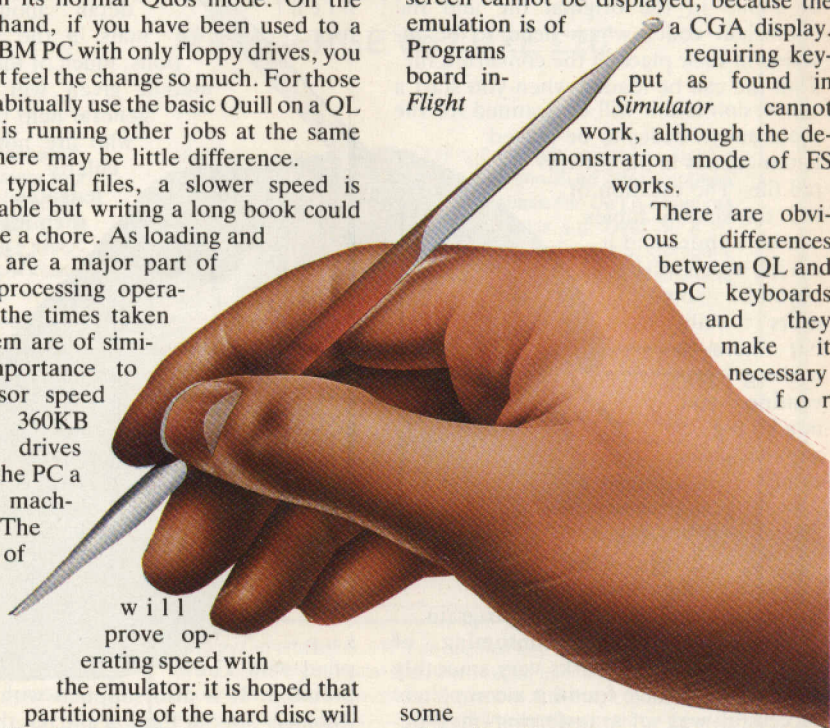
There are obvious differences between QL and PC keyboards and they make it necessary for

some PC keying to be altered to suit the QL. F1-F5 are the same but F6-F10 become CTRL+6-CTRL+0; you can customise keying to your requirements if the basic set-up does not suit you.

The "chocolate" version of *The Solution* is complete with MS-DOS version 4.0, a debugged version which seems to work normally but is suitable only for the PC & XT, not the AT; it includes GW Basic but Microsoft supplies no documentation for it. A public domain Basic is supplied with the "vanilla" version. IBM Basic and BasicA do not run, nor do they on typical PC clones. The main thing is that the emulation is very solid — flickering screens, crashes and error messages are unlikely, provided you have followed a few basic procedures given in the comprehensive manual.

DiscOVER

This was reviewed in the February, 1988 issue of *QL World*. Briefly, it treats one QL disc drive as the host (Qdos) and the other as the "alien" (MS-DOS). Put an appropriate disc into either drive and you can convert the files on it into a form which can be read on the other type of system, the converted files being written to the other drive. This is primarily a disc format change. What the basic program does not do is ensure that special codes used in files for one program are changed to whatever codes are appropriate to



another program. This is not something peculiar to conversion between different types of computer, as it occurs also if you transfer Quill .doc files into *The Editor* on the QL.

The program offers a partial solution to this problem in the form of transfer files. Users of *The Editor* will grasp the idea easily; you have to make a translation table, listing the special codes which will be found in the original file and the equivalent codes which need to be inserted in their place in the converted file. That file can be loaded when you start a conversion and it will be scanned for the replacement codes to be placed automatically in the converted file. The creation of the translation tables can be a chore and it is a job which will be beyond some users' ability but there is nothing preventing you making any necessary

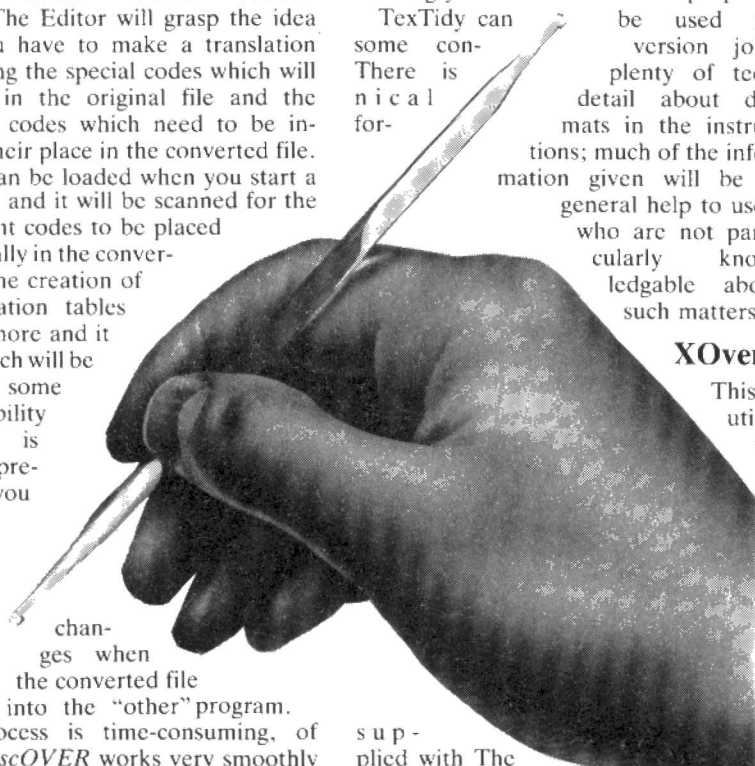
rather less intuitive, possibly because of the greater number of features available.

Neither program offers ready-made translation files for dealing with the code changes needed when moving from one program to another but advice is given on creating your own files for this purpose.

TextTidy can be used for some conversion jobs. There is plenty of technical detail about discs in the instructions; much of the information given will be of general help to users who are not particularly knowledgeable about such matters.

XOver

This is utility



changes when the converted file is loaded into the "other" program.

The process is time-consuming, of course. *DiscOVER* works very smoothly and fast and I have found it a completely successful way of transferring files between QL and PC. Transfers are not limited to QL-IBM but include the Apricot versions of IBM format. What you cannot do is take an Export-ed Abacus file, convert it, and have the formulae present when it is loaded into Abacus on a PC; the data is there but the formulae are not and this is a function of the Psion Export/Import process, not of *DiscOVER*.

Multi-DiscOVER

This is a development from *DiscOVER*. It allows transfers to and from a wider range of formats — IBM MS-DOS (including Apricot), BBC DFS/ADFS, CP/M and Unix CPIO. Transfers can be made to and from each of these and Qdos format. Automatic adjustment is made for drive type — 40- or 80-track — and 40-track discs can be read and normally written to in 80-track drives. Directory/sub-directory structures are supported. As with *TextTidy*, there is incompatibility with *Lightning*, *SpeedScreen* and *Toolkit III*, so that the program has to be run after a computer re-set. The menu screen and controls are similar to those for *DiscOVER*.

All QL storage devices, including hard disc, are usable; Thor models are supported. As with *DiscOVER*, the facility provided for looking at the contents of a file can be very useful. Users familiar with *Multi-DiscOVER*, although I find using it

supplied with *The Solution*. It is also supplied with *Media Manager Special Edition* and has the same menu format as that program. It enables files on MS-DOS or TOS (Atari) format discs to be transferred to Qdos format discs and vice versa. Other operations which are provided are Delete and Rename — DOS discs only — and conversion of the IBM version of ASCII character codes to the Qdos version; again, the reverse process is also available.

The latter function deals with the differences between the two company formats in handling characters with codes above decimal 127 and in the end-of-line codes used. Another facility is to format "alien" discs. You can take a Qdos-format disc and make it a DOS-format disc. An unexpected use for this was to format a 3.5in. disc with 40 tracks, since this odd format was required to attempt loading the Ant MS-DOS emulator; MS-DOS 3.2 and 3.3 on my AT refused to format this disc size with 40 tracks.

This was produced to solve some of the difficulties arising from use of *DiscOVER*. It is not a transfer utility in the same sense but performs the translation operations needed to make a text file look the same when transferred into a "new" program, whether on the QL or another computer. The program can remove any embedded codes from a file, producing pure ASCII output. In the process it can end lines with either line-feed (LF) or line-feed plus carriage-return (LF+CR) codes, as required by the receiving system.

The same operations can be performed on both PC Quill and QL/Thor Quill files. Likewise, WordStar files can be stripped of codes so that they are suitable for input and direct editing by *The Editor*, or for Importing into Quill. QL and PC Quill document files can be converted into a form suitable for use in WordStar, retaining the control codes for text enhancements such as bold and underlining.

The instructions warn the user that *TextTidy* has difficulty if run with *SpeedScreen*, *Lightning* or *Toolkit III* — it is satisfactory with *Toolkit II* — and this means you may have to run it in a "clean machine" rather than when your usual programs are loaded; this will be inconvenient for some users.

The screen presentation is consistent with the associated programs *discOVER* and *Multi-DiscOVER* and use is fairly straightforward without the need for the written instructions. For those who wish to transfer files between programs not dealt with, the principles of the process can be observed in *TextTidy*, for a relatively low price, to help with development of conversion files which can then be used to give automatic conversion with either version of *DiscOVER*.

SuperBasic C-Port

Originally called plain *Basic C-Port*, this program has been re-launched in improved form. Its purpose is to take source code — program lines — written in QL SuperBasic and convert them into code in the C language. The C code can then be compiled with a C compiler, which has to be ANSI- or Lattice C-compatible; a "Small C" compiler such as the DP Digital C is not fully-compatible.

Compilers for the QL, such as that from Metacomco, are no longer easily obtainable but PDQL expects to introduce a Lattice C compiler soon. As C is a relatively portable language, the code obtained from running SB code through *SuperBasic C-Port* on the QL can be compiled by a PC compiler and run on PCs; the British Standards Institute has cast some doubt on how compatible certain PC compilers are with the ANSI standard but it is not anticipated there will be any major problems compiling files from C-Port.

INFORMATION:

SuperBasic C-Port, £79; TextTidy, £10; Multi-DiscOVER, £39; DiscOVER £29.50:

PDQL Computer Systems, Unit 1, Heaton House, Camden Street, Birmingham B1 3BZ. Tel: 021 200 2313

The Solution MS-DOS emulator (including XOver), £79.95 (£129.95 with MS-DOS 4.0); Success CP/M emulator, £49.95:

Digital Precision, 222 The Avenue, Chingford, London E4 9SE. Tel: 01-527 5493.

I introduced functions to interrogate the channel definition block which stores details of windows and other channels in use a year ago. This month's code goes one step further, reading the QL display memory so that programs can check the colour of any point on the screen. This is a very useful function but curiously it is not available in popular commercial toolkits.

The code of the `PIXEL%` function illustrates many important features of QL design, including the arrangement of display memory, the way the console device can be extended and, for the first time in *DIY Toolkit*, the handling of default parameters.

Defaults are important because they allow programmers to simplify their code by omitting parameters. `PIXEL%` defaults to reading channel #1, which makes it an ideal companion for the `USE` command, introduced in *DIY Toolkit* in March last year. You can use a similar technique to add defaults to the channel access functions of your own toolkit code.

The `PIXEL%` function uses the `EXTOP` system-call to add extra code to the



QL display device. The workings of `EXTOP` were explained in the May, 1988 *DIY Toolkit*.

The first 32K of QL RAM is known as display memory, because any data stored in it appears as a pattern of dots on the screen. The first SuperBasic compiler, *Supercharge*, used screen memory to store details of the program it was compiling. Nowadays serious QL users have expanded memory and such tricks are unnecessary but *Supercharge* demonstrated that, apart from the mess on the screen display, memory works like normal RAM.

Dot control

Each consecutive pair of bytes or 'word' in display memory controls a row of dots on the QL screen. In the high-resolution `MODE 4` each word determines the colour of eight dots. The other mode trades extra colours for a reduced number of dots, so one word controls four dots in `MODE 8`.

DIY TOOLKIT

Each month Simon Goodwin adds new commands to the QL repertoire. This month he shows how to handle default parameters and read the colour of screen points.

`PIXEL%` can distinguish automatically between `MODE 8` and `MODE 4` displays. The difference in stored format is explained in the QL User Guide Memory Map in the Concepts section of the December, 1984 edition.

Dot colour

The function `PIXEL%` tells you the colour of a specified dot on the QL screen. The name of the function is a contraction of 'PIcture ELement' or 'PIcture CEL1', the technical name for the individual dots which make up text and graphics displays. The size of QL pixels varies depending on the graphics mode in use.

`MODE 8` displays are made up from a grid of 65,536 pixels — 256 vertically and the same number horizontally — where each dot can be in any one of eight colours, numbered from Black (0) to White (7), ranging through the spectrum in steadily-increasing brightness. `MODE 4` gives twice as many pixels — 131,072, made up of 256 lines with 512 pixels on each line.

Colours are numbered the same way regardless of mode but only four colours are allowed in `MODE 4`, so you can use two values for each displayed colour; 0 and 1 give black, 2 and 3 red, 4 and 5 green, and 6 and 7 white. Once a dot has been displayed `PIXEL%` cannot tell which variant you used and it makes no difference in any case, so in `MODE 4` it returns the values corresponding to the same colours in `MODE 8` — 0, 2, 4 and 7.

`PIXEL%` uses the QL 'pixel co-ordinate scheme', explained in the Concepts section of the QL User Guide. Co-ordinates are relative to the top left corner of the specified window, as in commands like `BLOCK`. To access the whole screen, just:

```
OPEN #3,SCR_512x256a0x0
```

and read from channel 3. You can read the colour of dots in a window whether or not

it is set up to allow character input, so `PIXEL%` works with `CONSOLE` or `ACREEN` windows.

The function has two or three parameters — an optional channel number, followed by the X and Y co-ordinates of the point to be read. It checks that the channel number corresponds to an open display channel normally an 'SCR' or 'CON' channel on a standard QL. You get a 'bad parameter' error if you specify the incorrect number of parameters or the wrong type of channel.

Like `BLOCK` and `WINDOW`, `PIXEL%` uses the same co-ordinate scheme regardless of mode. Horizontal co-ordinates in `MODE 8` are always even, so the 256 dots on a line are at co-ordinates 0, 2, 4 . . . 508, 510. If you supply an odd number it is rounded down.

Window mirror

Listing three shows how `PIXEL%` can be used to mirror the contents of a window. The procedure `MIRROR` in the listing uses `PIXEL%` and the `CHAN_W%` function from May last year to flip the contents of a window from left to right, so it can be read in a mirror.

You can use this procedure to generate projected displays with your QL, or when printing plans. It is particularly useful when building circuit boards; you can design the board from the component side and print a mirror-image of the finished layout when you need to know how everything will look from the solder side of the board.

This routine takes a time, even when compiled, because it uses the `BLOCK` routine which is optimised for large areas and it accesses each point in the window individually. `MIRROR` must make more than a quarter of a million system calls to flip a full-sized `MODE 4` window. Each call involves much work by the system ROM before the external routine is invoked. The `PIXEL%` code is trivial

* QL WORLD DIY TOOLKIT - pixel graphics function
 * Version 0.5, Copyright 1989 Simon N Goodwin.

```

*
start    lea.l    define,a1
         move.w   $110,a2      BP.INIT vector
         jmp      (a2)

*
* PIXELZ code - process 2 or 3 parameters
*
pixel    move.w   $112,a2      Vector to get integers
         jsr      (a2)
         bne.s    bad_exit
         moveq    #1,d0        Assume channel 1 for now
         subq.w   #2,d3        At least 2 parameters?
         beq.s    get_coords   Exactly 2, use #1
         bmi.s    bad_param    Less than 2: an error
         subq.w   #1,d3        Only one extra parameter?
         bne.s    bad_param    No, more than 3, complain
         move.w   0(a1,a6.l),d0 Get BASIC channel number
         addq.l   #2,a1        Discard channel parameter

*
get_coords move.w 0(a1,a6.l),d1 Get X co-ordinate
         move.w  2(a1,a6.l),d2 Get Y co-ordinate
         addq.l   #2,a1        Leave room for one INT
         move.l   a1,$58(a6)   Set maths stack pointer

*
* Check and convert channel number in D0 to ID in A0
*
chan_sel mulu     #40,d0        Channel table size
         add.l    $30(a6),d0    Add base offset
         cmp.l    $34(a6),d0
         bge.s    what_chan    Past end of table?
         move.l   0(a6,d0.l),d0
         bpl.s    chan_open    Negative if closed

*
* Error return points
*
what_chan moveq    #-6,d0        CHANNEL NOT OPEN error
bad_exit  rts
bad_param moveq    #-15,d0       BAD PARAMETER error
         rts
range_err moveq    #-4,d0        Out of Range report code
         rts

*
* Call EXTOP routine passing A0, D1 and D2
*
chan_open move.l   d0,a0        A0 is channel ID
         lea.l    pixtop,a2     Address of routine
         moveq    #-1,d3        Allow infinite time
         moveq    #9,d0         SD.EXTOP key
         trap     #3            Call the device driver
         move.l   $58(a6),a1     Retrieve maths stack pointer

*
return_int move.w  d1,0(a1,a6.l) Put result in space
         moveq    #3,d4        Indicate type is INT
         rts                  Return EXTOP error code

*
* Pixel eXTended Operation routine: reads pixel data.
* D1.W is the X co-ordinate, D2.W is the Y co-ordinate
* The result is returned in D1, or error code -4 in D0

```

```

*
pixtop   tst.w     d2            Check Y >= 0
         bmi.s     range_err
         cmp.w     30(a0),d2     Check Y < CH.HEIGHT
         bcc.s     range_err
         tst.w     d1            Check X >= 0
         bmi.s     range_err
         cmp.w     28(a0),d1     Check X < CH.WIDTH
         bcc.s     range_err
         add.w     24(a0),d1     Add window offset to X
         add.w     26(a0),d2     Add window offset to Y

*
* Find the relevant word in video memory
*
mode4    move.l    50(a0),a2     Get screen base address
         lsl.w     #7,d2         1 line uses 2^7 = 128 bytes
         add.w     d2,a2         Get address of start of line
         move.w    d1,d2         Save original X co-ordinate
         lsr.w     #2,d1         Get word offset on line
         and.w     #126,d1       Ensure offset is even, 0-126
         add.w     d1,a2         A2 -> relevant screen word
         move.w    (a2),d1       D1.W is the video word
         and.w     #7,d2         D2 = pixel offset in word, 0-7

*
* Extract the pixel in mode 4 or 8
*
         btst      #3,$52(a6)    Check mode
         bne.s     mode8         Use mask for 8 colours
         moveq     #7,d0
         sub.w     d2,d0         Reverse offset to 7-0
         lsr.w     d0,d1         Move pixel data to LS bits
         and.w     #257,d1       Mask for 4 colours
         add.b     d1,d1         Double red bit weight
         move.w    d1,d2         D2 = 00000006 00000000
         lsr.w     #6,d2         D2.B = 00000600
         or.b      d2,d1         D1.B = 00000600
         cmp.b     #6,d1         Conventional value 0, 2, 4 ?
         bne.s     got_value
         addq.b    #1,d1         Translate 6 to 7 (White)
         bra.s     got_value

*
mode8    moveq     #6,d0
         bclr      #0,d2         Ensure even X in MODE 8
         sub.w     d2,d0         Reverse offset to 6-0
         lsr.w     d0,d1         Move pixel data to LS bits
         and.w     #515,d1       Mask for 8 colours
         move.w    d1,d2         D2 = 00000060 00000000
         lsr.w     #7,d2         D2.B = 00000600
         or.b      d2,d1         D1.B = 00000600

*
got_value ext.w     d1
         moveq     #0,d0         Signal no error
         rts

*
define   dc.w      0,0          No procedures
         dc.w      1            One function
         dc.w      pixel-#
         dc.b      6,'PIXELZ'
         ds.w      0
         dc.w      0            End of list
         end

```

compared to the BLOCK routine and the unavoidable Sinclair TRAP handler.
 For best results with Turbo add IMPLICIT% x,y at the head of the program. If you want the code to run really fast you must write a new code to process several dots at once.

Listing three will not work in MODE 8 without changes, because of the smaller number of pixels in that mode. The BLOCK 1,1 commands set a single MODE 4 pixel, which Sinclair rounds down to no MODE 8 pixels at all.
 To flip a window in MODE 8, change

lines 200 and 210 to start BLOCK 2,1 . . . and add STEP 2 at the end of line 180, so the program reads only even X co-ordinates. The good news is that the MODE 8 version will be twice as fast, because it has half as many pixels to read for a given display area.



You can use PIXEL% for many other applications when speed is less of a problem — for instance to check when moving graphics collide on the screen.

No MODE 12

The functions have been written to work reliably on any QL or compatible, including the CST Thor and Thor 16. They do not handle the Thor 16-colour MODE 12, because designer David Oliver failed to supply information about Thor-specific features, but it should be easy to adapt the MODE 8 code to suit MODE 12 if you can obtain the information from CST.

Like the CHAN functions, PIXEL% uses the SD.EXTOP routine to access channel details. That means it is not disturbed by the extra channel information which QRAM and Thor windowing systems tack on to the start of a channel block. QLs and most Thors keep the display information at a fixed address, 131072, but the Thor 16 is more flexible and can have several 32K display maps at different addresses. PIXEL% reads the base address of the display from offset 50 in the channel details, so it takes multiple display areas in its stride.

The code for the channel access functions is listed in two forms. Listing two gives you a quick way to enter the code without using an assembler. It loads the equivalent machine code from DATA statements and saves the code in a file. Once you have loaded that file, as follows, you can use PIXEL% in your own programs:

```
base=RESPR(242) : LBYTES "file
name", base : CALL base
```

Type NEW after loading the code on an AH or JM QL to ensure that early QL ROMs are not confused by prior use of the name PIXEL%.

Loader

The first part of listing two is Marcus Jeffery's standard loader, used in every month's DIY Toolkit project. Only the DATA, from line 590 onwards, changes from month to month. Listing one is the assembly code program, assembled using

DIY Toolkit June 1989, listing 2.

```
100 REMark Sinclair QL World HEX LOADER
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
150 CLS: RESTORE : READ space: start=RESPR(space)
160 PRINT "Loading Hex..." : HEX_LOAD start
170 INPUT "Save to file...";f$
180 SBYTES f$,start,byte : STOP
190 :
200 DEFine FuNction DECIMAL(x)
210 RETURN CODE(h$(x))-48-7*(h$(x)>"9")
220 END DEFine DECIMAL
230 :
240 DEFine PROCedure HEX_LOAD(start)
290 byte = 0 : checksum = 0
300 REPEAT load_hex_digits
310 READ h$
320 IF h$="*" : EXIT load_hex_digits
330 IF LEN(h$) MOD 2
340 PRINT"Odd number of hex digits in: ";h$
350 STOP
360 END IF
370 FOR b = 1 TO LEN(h$) STEP 2
380 hb = DECIMAL(b) : lb = DECIMAL(b+1)
390 IF hb<0 OR hb>15 OR lb<0 OR lb>15
400 PRINT"Illegal hex digit in: ";h$ : STOP
420 END IF
430 POKE start+byte,16*hb+lb
440 checksum = checksum + 16*hb + lb
450 byte = byte + 1
460 END FOR b
470 END REPEAT load_hex_digits
480 READ check
490 IF check <> checksum
500 PRINT"Checksum incorrect. Recheck data.":STOP
520 END IF
530 PRINT"Checksum correct, data entered at: ";start
560 END DEFine HEX_LOAD
570 :
580 REMark Space requirements for the machine code
590 DATA 242
600 :
610 REMark Machine code data
620 DATA "43FA00DE34790000","01104ED234790000"
630 DATA "01124E9266367001","5543670C6B305343"
640 DATA "662C3031E8005489","3231E8003431E802"
650 DATA "54892D490058C0FC","0028D0AE0030B0AE"
660 DATA "00346C0620360800","6A0C70FA4E7570F1"
670 DATA "4E7570FC4E752040","45FA001476FF7009"
680 DATA "4E43226E00583381","E80078034E754A42"
690 DATA "6BE0B468001E64DA","4A416BD6B268001C"
700 DATA "64D0D2680018D468","001A24680032EF4A"
710 DATA "D4C23401E4490241","007ED4C132120242"
720 DATA "0007082E00030034","661C70079042E069"
730 DATA "02410101D2013401","EC4A8202B23C0006"
740 DATA "6618520160147006","088200009042E069"
750 DATA "024102033401EE4A","8202488170004E75"
760 DATA "000000000001FF26","06504958454C2500"
770 DATA "0000","*",19106
```

HiSoft DevPac. You can type this text into your assembler if you want to customise the code or merge it with other routines.

The START routine calls BP.INIT, the ROM vector which adds new commands to SuperBasic. The table labelled DEFINE, at the end of the listing, indicates the name and address of the PIXEL routine.

The first step is to read the parameters — the optional channel number and two

co-ordinates. They are all integers, so we call CA.GTINT to fetch them from the place indicated by A3 and A5. GTINT returns with a non-zero result, signifying a 'bad parameter' or 'error in expression' if it cannot find suitable values. Otherwise it puts them on the maths stack, pointed to by A1 offset from A6, with the first parameter at the lowest address on the stack.

The total number of parameters fetched

DIY Toolkit June 1989, listing 3.

```
100 REMark MODE 4 Window reflecting routine
110 REMark QL World DIY Toolkit June 1989
120 REMark Copyright 1989 Simon N Goodwin
130 :
140 DEFine PROCedure MIRROR(ch%)
150 LOCAL max_x%,y,x,dot%
160 max_x%=CHAN_W%(ch%,28)-1
170 FOR y=0 TO CHAN_W%(ch%,30)-1
180 FOR x=0 TO max_x% DIV 2
190   dot%=PIXEL%{#ch%,max_x%-x,y)
200   BLOCK 1,1,max_x%-x,y,PIXEL%{#ch%,x,y)
210   BLOCK 1,1,x,y,dot%
220 END FOR x
230 END FOR y
240 END DEFine MIRROR
```

is returned in D3. If that is fewer than two or more than three, we get a 'bad parameter' error. If we have three parameters the first is a channel number; otherwise we assume channel #1. The parameters end up in registers D0, D1 and D2 and the maths stack is adjusted to leave room for a single integer result.

Not open

The familiar CHAN_SEL code converts a Basic channel number into an internal system identifier, by looking through the SuperBasic table of channel details. If the identifier is negative, or the entry would be outside the limits of the table, the code reports 'channel not open'.

Next we call SD.EXTOP, which lets us add new code to a display device. The

ROM code invoked by the TRAP #3 converts the channel identifier in A0 into the address of the first documented part of the channel block. Only screen and console channels recognise EXTOP; others give a 'bad parameter' error.

It is possible that another task is already using the channel, in which case Qdos cannot use SD.EXTOP immediately as only one task can use a particular channel at any time. The timeout value -1 in D3 ensures that the system keeps trying to perform the operation every time tasks are swapped, until it succeeds because the other task has finished with the channel.

This is called an 'infinite' timeout, with good reason, but should not cause problems; if necessary you can set up a separate window overlaying the others, just for PIXEL% calls.

Eventually the console driver calls the routine pointed to by A2, the code which

finds and interrogates a pixel. PIXEL% uses D1 and D2 to pass the co-ordinates and D1 to read the resultant colour. A1 is also passed back and forth by EXTOP but PIXEL% does not use it. The original value of D2 is reinstated when the call is complete.

Pixtop

The PIXTOP routine starts by checking that the co-ordinates fit inside the window; then it adds the window offsets, making the co-ordinates relative to the top left corner of the screen.

The next step is to find the word of video memory which stores the pixel colour. Each line takes 128 bytes, so the start of memory for the relevant line of pixels is found by adding Y% * 128 to the start address of video memory, taken from the channel details. There are 512 possible X co-ordinates on each 64-word line. We find the required word by converting the pixel co-ordinate, 0-511, into an even offset between 0 and 126.

The LSR.W divides D1 by 4 quickly, while the AND.W ensures an even result. The last three bits of the X co-ordinate tell us the location of the pixel data inside the word and end up in D2. Later the value is

'Defaults are important, because they simplify code by omitting parameters.'

'reversed', as solid colour numbers use the lowest bits of a byte, whereas pixel offset 0 corresponds to the most significant bits in a video word, which need to be shifted most.

Bit #3 of the system variable SV.MCSTA is set if we are in MODE 8; it is easy to check this, as A6 points to the system variables when an EXTOP routine is called. The 'flash' bit is ignored in MODE 8, as flashing is little-used and you cannot work out whether or not a particular pixel is flashing without scanning the previous part of the line. A set flash bit affects all the pixels up to the end of the line or the next set flash bit.

The code for each mode uses a succession of shifts to extract the details of one pixel and convert it into a standard colour number. The comments alongside the code show the way red, green and blue colour information is shuffled. A successful call returns with the result in D1 and zero in D0.

● Next month I will have more code and commentary. Please send your suggestions if you would like me to explore a specific area in this column, or implement new commands - particularly ones unavailable in commercial toolkits.

QL World DIY Toolkit PIXEL% demonstration.

```
100 REMark MODE 4 Window reflecting routine
110 REMark QL World DIY Toolkit June 1989
120 REMark Copyright 1989 Simon N Goodwin
130 :
140 DEFine PROCedure MIRROR(ch%)
150 LOCAL max_x%,y,x,dot%
160 max_x%=CHAN_W%(ch%,28)-1
170 FOR y=0 TO CHAN_W%(ch%,30)-1
180 FOR x=0 TO max_x% DIV 2
190   dot%=PIXEL%{#ch%,max_x%-x,y)
200   BLOCK 1,1,max_x%-x,y,PIXEL%{#ch%,x,y)
210   BLOCK 1,1,x,y,dot%
220 END FOR x
230 END FOR y
240 END DEFine MIRROR
```


Quanta shows the hard discs

Simon Goodwin dons a pair of his hats at Northampton.

Hard discs were the dominant theme at the latest Quanta QL User Group workshop at Northampton; three systems were on show, with several others promised. David Richards was demonstrating the official Quanta prototype disc and interface. It runs up to four IBM-type hard discs or floppy drives and uses the M212 transputer intelligent disc controller. A 10MB system costs around £350.

The Miracle Systems £399 Winchester drive was on display, running Tony Tebby control software and having a capacity of almost 32MB. It uses an IBM filecard internally but improves on the reliability of a PC with an auto-park routine. It moves the vibration-sensitive disc head safely out of harm's way if the drive is not accessed for five seconds or more.

That reduces the risk of a data-scrambling head crash but even then Winchester users would be wise to buy a copy of

the PDQL HardBack back-up utility; 32MB is a great deal of re-typing if the worst happens.

The surprise arrival was a second commercial hard disc system from Rebel Electronics of York. Its RB-100 interface works with one or two standard Winchester drives, each with a capacity of 20, 40, 60 or 80MB.

The interface ROM includes in-house disc control software and a faster, more flexible version of *Speedscreen*, the QL text accelerator. An 8K sector buffer on the board gives extra disc speed.

The Rebel hard disc interface plugs into the main QL expansion connector. It uses reliable but expensive Western Digital chips and costs slightly less than £200, plus the cost of the drive. Rebel launched simultaneously a much-needed expansion backplane which fans out the QL expansion connector, allowing up to four peripheral cards to plug in at once.

The £83 RB-50E includes fast buffers, firmware to link



Tony Tebby of QJump addresses the multitude.

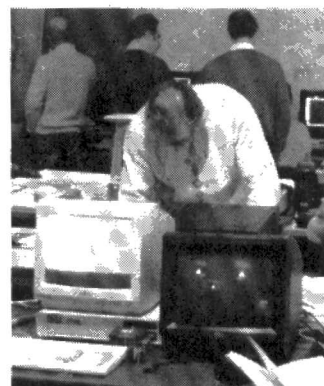
devices into Qdos and a socket for a second power supply, in case you decide to cascade power-hungry peripherals. A complete expansion of Winchester interface, four-way backplane and 80MB drive costs £699, or £664 to Quanta members.

As usual, the workshop featured presentations from a succession of QL luminaries, including John Silk of PDQL and Freddy Vachha of Digital Precision. Tony Tebby led the way with a talk on his *QJump* products. It seems that work on the Miracle hard disc has rekindled his enthusiasm for the QL, although *GRAM 2* is still far from complete.

Creative CodeWorks boss and *QL World* contributor Simon Goodwin discussed work in progress, including a secret weapon due for launch in the autumn. He rounded off his presentation with a personal impression of the quirks of magazine publishing, from his perspective as the compiler of *DIY Toolkit*, and later had to endure an interrogative telephone call from the editor on the strength of this.

In the next room hardware and communications specialist Tony Firshman slaved over a hot soldering iron, fitting the Futura Datacentre QL emulator into a succession of Atari STs. A plethora of QL spares was available from him and Dennis Briggs of Quanta in the packed machine room upstairs.

Former Sinclair and *QJump* programmer Jonathan Oakley



Thor in his glory.

discussed Tyche, the ill-fated Sinclair QL follow-up, and his plans to build a professional Midi synthesiser interface based on the 68681 dual UART chip. The interface is the easy part; it is the device driver and Midi editor which will need a good deal of work. Oakley wrote *GRAM*, so he has the experience to make a good job of a QL Midi device.

This was the second QL workshop at Northampton and more than 200 QL enthusiasts appeared during the weekend. Quanta organisers plan another workshop in the same area in the autumn. The QL scene is buzzing at the moment, with Quanta members at the forefront and prospects for new hardware and software in 1989 look excellent.

Further information about Quanta, the Independent QL User Group, is available from the membership secretary, Philip Borman, at 15 Grosvenor Crescent, Grimsby DN32 0QJ.



Freddy Vaccha of Digital Precision joins in.



Johnathan Oakley (QJump) and Simon Goodwin compute.

THE

P + R : O = G < S

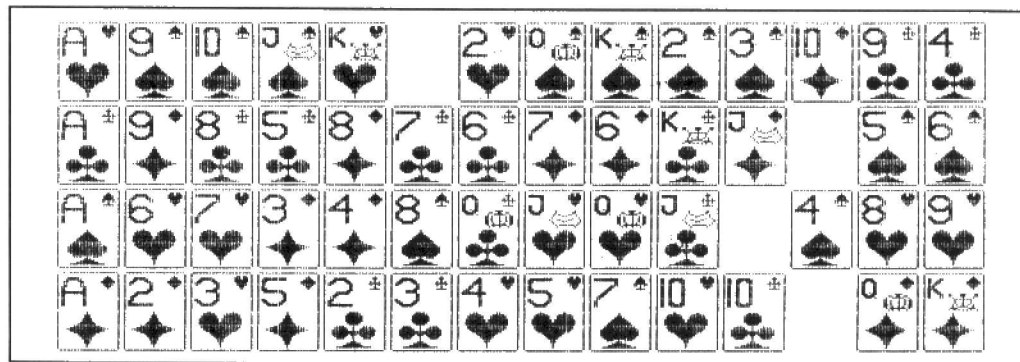
If you have a program worthy of consideration, send it to 'The Progs',
Sinclair QL World, Greencoat House, Francis Street, London SW1P 1DG.
We pay for everything published at the usual page rates.

Program of the month

CRAZY CARDS by Quintens Pierre

The aim of the game is to place each card in increasing order after the corresponding ace. To do this you must move the appropriate card into a free place, so that the card on its left is the same colour and one value below the card moving in. This is repeated until it is impossible to move further. Then the program will reshuffle the unused cards for the next round. There is a maximum of four rounds.

Use the left, right, up and down keys to move the vertical and horizontal cursors; use the



space bar to move cards. The F4 key toggles between a default mode — the indicated free place is filled by the right card — and an alternative — the space bar moves the indi-

cated card to the correct free place. F5 reshuffles the cards. Exit from the game is via escape.

It may look simple but it is extremely addictive.

We are not able to offer Crazy Cards on the Microdrive Exchange this month, although we hope to do so in future. Please could the author write to us?

```

10 CLS
20 PRINT 'CRAZY CARDS'
30 PRINT 'BY QUINTENS PIERRE'
40 PRINT '1 rue Henneumont'
50 PRINT '5198 ANHEE'
60 PRINT 'BELGIUM'
70 PRINT 'PHONE 082/612677'
80 AD=RESPR(6144)
90 LBYTES MDV1_CRAZY_CDE,AD
100 CALL AD

10 REMark          CRAZY-CARDS
20 REMark          BY
30 REMark          QUINTENS PIERRE
40 REMark          COPYRIGHT 87

100 DIM CK(18)
110 AD=RESPR(0)
120 IF RESPR(0)=262144 THEN AD=RESPR(6144)
130 FOR F=1 TO 18
140 CK(F)=0
150 FOR G=0 TO 191
160 READ D
170 CK(F)=CK(F)+D
180 POKE AD+(F-1)*192+G,D
190 NEXT G
200 NEXT F
220 RESTORE 10000
230 FOR F=1 TO 18
240 READ N
250 IF CK(F)<>N THEN
260 PRINT 'ERROR AFTER LINE ';
```

```

270 PRINT 1000+(F-1)*240
280 END IF
290 NEXT F
300 SBYTES MDV1_CRAZY_C,AD,3456
310 PRINT 'NOW TYPE : '\
320 PRINT ' - NEW;'
330 PRINT ' - LBYTES MDV1_CRAZY_C,RESPR(0);'
340 PRINT ' - CALL RESPR(0). '
350 STOP

1000 DATA 56,120, 0,198, 67,250, 0,196
1010 DATA 78,148, 67,250, 0,184, 34,136
1020 DATA 112, 16,114, 0,116, 1, 78, 65
1030 DATA 75,250, 1, 56,116, 13, 66,157
1040 DATA 81,202,255,252, 73,250, 1, 36
1050 DATA 40,188, 13, 27, 41, 55, 73,250
1060 DATA 2,210, 40,188, 0, 0, 0, 0
1070 DATA 73,250, 4,116, 40,188, 0, 0
1080 DATA 0, 0,112, 32,118,255, 78, 67
1090 DATA 112, 45,114, 2,116, 1, 78, 67
1100 DATA 97, 0, 0,132, 32,122, 0,110
1110 DATA 112, 16,114, 9,116, 11,118,255
1120 DATA 78, 67,112, 7,116, 10,118,255
1130 DATA 67,250, 2,156, 78, 67,112, 16
1140 DATA 114, 17,116, 11, 78, 67,112, 5
1150 DATA 34,122, 2,136, 34, 17, 6, 1
1160 DATA 0, 49,118,255, 78, 67, 75,249
1170 DATA 0, 2,104, 16,124, 51,126, 7
1180 DATA 58,188,255,255,219,252, 0, 0
1190 DATA 0,128, 81,207,255,244,155,252
1200 DATA 0, 0, 3,254, 81,206,255,232
1210 DATA 75,249, 0, 2, 4,122, 60, 60
```



```

1220 DATA 0,191, 58,188,255,255,219,252
1230 DATA 0, 0, 0,128, 81,206,255,244
1240 DATA 96, 0, 1, 18, 0, 0, 0, 0
1250 DATA 0, 0,246, 2,144, 50, 2, 0
1260 DATA 1, 0, 0, 0, 0, 0, 75,250
1270 DATA 0,122,122, 55, 12, 5, 0, 0
1280 DATA 109, 0, 0,100, 12, 21, 0, 0
1290 DATA 103, 0, 0, 10, 82,141, 83, 5
1300 DATA 96, 0,255,234, 73,250, 0, 84
1310 DATA 124, 3,186, 28,103, 0, 0, 10
1320 DATA 81,206,255,248, 96, 0, 0, 10
1330 DATA 82,141, 83, 5, 96, 0,255,206
1340 DATA 50, 57, 0, 2,128, 46, 52, 1
1350 DATA 238, 90,213,121, 0, 2,128, 46
1360 DATA 2,129, 0, 0, 0, 63, 82, 1
1370 DATA 12, 1, 0, 52, 98, 0,255,226
1380 DATA 71,250, 0, 32,126, 55,178, 27
1390 DATA 103, 0,255,214, 81,207,255,248
1400 DATA 26,193, 93,205,255,152, 96, 0
1410 DATA 0, 70, 0, 0, 0, 0, 0, 0
1420 DATA 0, 0, 0, 0, 0, 0, 0, 0
1430 DATA 0, 0, 0, 0, 0, 0, 0, 0
1440 DATA 0, 0, 0, 0, 0, 0, 0, 0
1450 DATA 0, 0, 0, 0, 0, 0, 0, 0
1460 DATA 0, 0, 0, 0, 0, 0, 0, 0
1470 DATA 0, 0, 0, 0, 0, 0, 0, 0
1480 DATA 0, 0, 0, 0, 0, 0, 0, 0
1490 DATA 0, 0, 0, 0, 0, 0, 73,250
1500 DATA 255,194,124, 55, 18, 28, 97, 0
1510 DATA 5, 4, 71,250, 0, 32, 72,227
1520 DATA 2, 8, 97, 0, 5, 68, 71,250
1530 DATA 0, 12, 76,219, 16, 64, 81,206
1540 DATA 255,228, 78,117, 0, 0, 0, 0
1550 DATA 0, 0, 0, 0, 0, 0, 0, 0
1560 DATA 0, 0, 0, 0, 0, 0, 0, 0
1570 DATA 0, 0, 0, 0, 0, 0, 0, 0
1580 DATA 0, 0, 0, 0, 66,134, 66,135
1590 DATA 75,250, 2,212, 60, 29, 62, 21
1600 DATA 48, 60,255, 0, 97, 0, 2,146
1610 DATA 97, 0, 2,172, 73,250,255,100
1620 DATA 120, 55,124, 55, 71,250,255, 84
1630 DATA 38,188, 13, 27, 41, 55, 71,250
1640 DATA 255, 74,118, 3,188, 27,103, 0
1650 DATA 0, 10, 81,203,255,248, 96, 0
1660 DATA 0, 8, 82,140, 4, 6, 0, 1
1670 DATA 18, 28, 12, 1, 0, 1,103, 0
1680 DATA 0, 34, 12, 1, 0, 14,103, 0
1690 DATA 0, 26, 12, 1, 0, 27,103, 0
1700 DATA 0, 18, 12, 1, 0, 40,103, 0
1710 DATA 0, 10, 81,206,255,194, 96, 0
1720 DATA 0, 70,118, 55,150, 4, 71,250
1730 DATA 255, 10, 23,129, 48, 0, 71,250
1740 DATA 255,112, 72,227, 10, 8, 25, 60
1750 DATA 0, 0, 28, 4, 97, 0, 4, 62
1760 DATA 97, 0, 4,134, 71,250,255, 78
1770 DATA 76,219, 16, 80,114, 0, 97, 0
1780 DATA 4, 44, 97, 0, 4,116, 71,250
1790 DATA 255, 60, 76,219, 16, 80, 4, 4
1800 DATA 0, 14, 96, 0,255,182, 73,250
1810 DATA 254,202,124, 55,126, 0, 18, 28
1820 DATA 12, 1, 0, 13,103, 0, 0, 42
1830 DATA 12, 1, 0, 26,103, 0, 0, 34
1840 DATA 12, 1, 0, 39,103, 0, 0, 26
1850 DATA 12, 1, 0, 52,103, 0, 0, 18
1860 DATA 81,206,255,220, 12, 7, 0, 4
1870 DATA 109, 0, 0,226, 96, 0, 0, 36
1880 DATA 122, 3, 12, 6, 0, 0,111, 0
1890 DATA 255,232, 12, 20, 0, 0,102, 0
1900 DATA 255,224, 82, 7, 82,140, 4, 6
1910 DATA 0, 1, 81,205,255,230, 96, 0
1920 DATA 255,208, 73,250,254,110,122, 3
1930 DATA 124, 11, 18, 28, 82, 1,178, 20
1940 DATA 102, 0, 0, 30, 81,206,255,244
1950 DATA 84,140, 81,205,255,236, 96, 0
1960 DATA 2,194, 0, 0, 0, 0, 65, 84
1970 DATA 84, 69, 77, 80, 84, 32, 58, 32
1980 DATA 112, 16,114, 17,116, 11, 32,122
1990 DATA 253,172, 78, 67,112, 5, 67,250
2000 DATA 255,226, 6,145, 0, 0, 0, 1
2010 DATA 34, 17, 12, 1, 0, 4,108, 0
2020 DATA 2,242, 6,129, 0, 0, 0, 49
2030 DATA 118,255, 32,122,253,136, 78, 67
2040 DATA 75,250,254, 8, 73,250,254, 13
2050 DATA 120, 54, 97, 0, 0, 40, 73,250
2060 DATA 254, 17,120, 40, 97, 0, 0, 30
2070 DATA 73,250,254, 21,120, 26, 97, 0
2080 DATA 0, 20, 73,250,254, 25,120, 12
2090 DATA 97, 0, 0, 10, 97, 0,253,104
2100 DATA 96, 0, 0, 42,116, 12, 18, 20
2110 DATA 4, 1, 0, 1,178, 44,255,255
2120 DATA 102, 0, 0, 14, 82,140, 4, 4

```

```

2130 DATA 0, 1, 81,202,255,234, 78,117
2140 DATA 26,196, 24,252, 0, 0, 81,202
2150 DATA 255,250, 78,117,112, 1,118,255
2160 DATA 32,122,253, 34, 78, 67,126, 0
2170 DATA 48, 60,255,255, 75,250, 1, 0
2180 DATA 44, 21, 62, 6, 66, 70, 72, 70
2190 DATA 12, 1, 0,244,102, 0, 0, 22
2200 DATA 67,250,253, 6, 74, 81,103, 0
2210 DATA 0, 8, 66, 81, 96, 0, 0, 6
2220 DATA 50,188, 0, 1, 12, 1, 0,248
2230 DATA 103, 0,255, 8, 12, 1, 0, 27
2240 DATA 103, 0, 2,174, 12, 1, 0, 32
2250 DATA 103, 0, 0,204, 12, 1, 0,192
2260 DATA 102, 0, 0, 22, 12, 70, 0, 0
2270 DATA 103, 0,255,162, 97, 0, 0,122
2280 DATA 4, 70, 0, 8, 96, 0, 0, 82
2290 DATA 12, 1, 0,200,102, 0, 0, 22
2300 DATA 12, 70, 0, 96,103, 0,255,134
2310 DATA 97, 0, 0, 94, 6, 70, 0, 8
2320 DATA 96, 0, 0, 54, 12, 1, 0,208
2330 DATA 102, 0, 0, 22, 12, 71, 0, 0
2340 DATA 103, 0,255,106, 97, 0, 0, 96
2350 DATA 4, 71, 24, 0, 96, 0, 0, 26
2360 DATA 12, 1, 0,216,102, 0,255, 86
2370 DATA 12, 71, 72, 0,103, 0,255, 78
2380 DATA 97, 0, 0, 68, 6, 71, 24, 0
2390 DATA 58,198, 58,135, 48, 60,255, 0
2400 DATA 12, 1, 0,200,110, 0, 0, 10
2410 DATA 97, 0, 0, 14, 96, 0,254, 24
2420 DATA 97, 0, 0, 36, 96, 0,254, 16
2430 DATA 73,249, 0, 2,104, 16,217,198
2440 DATA 116, 7,118, 3, 56,192, 81,203
2450 DATA 255,252,217,252, 0, 0, 0,120
2460 DATA 81,202,255,240, 78,117, 73,249
2470 DATA 0, 2, 4,122,217,199,116, 47
2480 DATA 56,128,217,252, 0, 0, 0,128
2490 DATA 81,202,255,246, 78,117, 0, 0
2500 DATA 0, 0, 0, 0, 0, 0,140,252
2510 DATA 0, 8,142,252, 24, 0, 2,134
2520 DATA 0, 0, 0,255, 2,135, 0, 0
2530 DATA 0,255, 82, 6,206,252, 0, 14
2540 DATA 222, 6, 75,250,251,244, 74, 85
2550 DATA 102, 0, 0,128, 75,250,252,116
2560 DATA 12, 53, 0, 0,112, 0,102, 0
2570 DATA 254,180, 18, 53,112,255, 12, 1
2580 DATA 0, 0,103, 0,254,168, 12, 1
2590 DATA 0, 13,103, 0,254,160, 12, 1
2600 DATA 0, 26,103, 0,254,152, 12, 1
2610 DATA 0, 39,103, 0,254,144, 12, 1
2620 DATA 0, 52,103, 0,254,136, 82, 1
2630 DATA 124, 55,156, 7, 71,250,252,162
2640 DATA 72,227, 65, 4, 97, 0, 1,118
2650 DATA 97, 0, 1,190, 71,250,252,134
2660 DATA 76,219, 32,130, 73,250,252, 28
2670 DATA 124, 55,178, 28,103, 0, 0, 6
2680 DATA 81,206,255,248, 25, 60, 0, 0
2690 DATA 27,129,112, 0, 97, 0, 1, 78
2700 DATA 114, 0, 97, 0, 1,148, 96, 0
2710 DATA 253, 46, 75,250,251,246, 12, 53
2720 DATA 0, 0,112, 0,103, 0,254, 54
2730 DATA 18, 53,112, 0, 73,250,251,228
2740 DATA 124, 54, 4, 1, 0, 1,178, 28
2750 DATA 103, 0, 0, 6, 81,206,255,248
2760 DATA 12, 20, 0, 0,102, 0,254, 22
2770 DATA 82, 1, 27,188, 0, 0,112, 0
2780 DATA 71,250,252, 34, 72,211, 32,192
2790 DATA 126, 55,158, 6, 27,129,112, 0
2800 DATA 78,186, 0,250, 78,186, 1, 66
2810 DATA 71,250,252, 10, 76,211, 32,192
2820 DATA 114, 0,124, 55,156, 7, 97, 0
2830 DATA 0,228, 97, 0, 1, 44, 96, 0
2840 DATA 252,198, 32,122,251, 0,112, 32
2850 DATA 118,255, 78, 67, 67,250, 0, 14
2860 DATA 116, 70,118,255,112, 7, 78, 67
2870 DATA 96, 0, 0,168, 10, 10, 10, 10
2880 DATA 32, 32, 32, 32, 32, 32, 32, 32
2890 DATA 32, 32, 32, 32, 32, 67, 79, 78
2900 DATA 71, 82, 65, 76, 85, 84, 65, 84
2910 DATA 73, 79, 78, 83, 32, 10, 10, 10
2920 DATA 32, 32, 32, 32, 32, 32, 32, 80
2930 DATA 82, 69, 83, 83, 32, 65, 78, 89
2940 DATA 32, 75, 69, 89, 32, 84, 79, 32
2950 DATA 80, 76, 65, 89, 32, 65, 71, 65
2960 DATA 73, 78, 32,122,250,160,112, 32
2970 DATA 118,255, 78, 67, 67,250, 0, 14
2980 DATA 116, 70,118,255,112, 7, 78, 67
2990 DATA 96, 0, 0, 72, 10, 10, 10, 10
3000 DATA 32, 32, 32, 32, 32, 32, 32, 32
3010 DATA 32, 32, 32, 83, 79, 82, 82, 89
3020 DATA 32, 66, 85, 84, 32, 89, 79, 85
3030 DATA 32, 70, 65, 73, 76, 10, 10, 10

```



```

3040 DATA 32, 32, 32, 32, 32, 32, 32, 80
3050 DATA 82, 69, 83, 83, 32, 65, 78, 89
3060 DATA 32, 75, 69, 89, 32, 84, 79, 32
3070 DATA 80, 76, 65, 89, 32, 65, 71, 65
3080 DATA 73, 78, 112, 1, 118, 255, 32, 122
3090 DATA 250, 60, 78, 67, 96, 0, 249, 138
3100 DATA 32, 122, 250, 50, 112, 2, 78, 66
3110 DATA 66, 128, 78, 117, 126, 55, 158, 6
3120 DATA 75, 249, 0, 2, 4, 8, 12, 7
3130 DATA 0, 14, 109, 0, 0, 52, 12, 7
3140 DATA 0, 28, 108, 0, 0, 14, 4, 7
3150 DATA 0, 14, 219, 252, 0, 0, 24, 0
3160 DATA 96, 30, 12, 7, 0, 42, 108, 0
3170 DATA 0, 14, 4, 7, 0, 28, 219, 252
3180 DATA 0, 0, 48, 0, 96, 10, 4, 7
3190 DATA 0, 42, 219, 252, 0, 0, 72, 0
3200 DATA 207, 252, 0, 8, 219, 199, 78, 117
3210 DATA 16, 1, 66, 129, 18, 0, 66, 128
3220 DATA 12, 1, 0, 0, 103, 0, 0, 248
3230 DATA 4, 1, 0, 1, 131, 252, 0, 13
3240 DATA 48, 1, 66, 65, 72, 65, 67, 250
3250 DATA 1, 4, 69, 250, 3, 112, 71, 250
3260 DATA 4, 44, 73, 250, 4, 184, 40, 1
3270 DATA 4, 132, 0, 0, 0, 10, 201, 252
3280 DATA 0, 48, 195, 252, 0, 48, 193, 252
3290 DATA 0, 48, 36, 0, 208, 128, 124, 23
3300 DATA 126, 1, 26, 177, 16, 0, 27, 113
3310 DATA 16, 0, 0, 1, 27, 114, 32, 0
3320 DATA 0, 4, 27, 114, 32, 0, 0, 5
3330 DATA 27, 116, 0, 0, 12, 0, 27, 116
3340 DATA 0, 0, 12, 1, 27, 116, 0, 2
3350 DATA 12, 4, 27, 116, 0, 2, 12, 5
3360 DATA 10, 85, 255, 255, 10, 109, 255, 255
3370 DATA 0, 4, 10, 109, 255, 255, 12, 0
3380 DATA 10, 109, 255, 255, 12, 4, 12, 64
3390 DATA 0, 144, 109, 0, 0, 26, 27, 124
3400 DATA 0, 255, 0, 1, 27, 124, 0, 255
3410 DATA 0, 5, 27, 124, 0, 255, 12, 1
3420 DATA 27, 124, 0, 255, 12, 5, 12, 68
3430 DATA 0, 0, 109, 0, 0, 54, 22, 45
3440 DATA 0, 4, 10, 3, 0, 255, 134, 51
3450 DATA 64, 0, 27, 67, 0, 4, 12, 64
3460 DATA 0, 144, 109, 0, 0, 18, 10, 45
3470 DATA 0, 255, 0, 4, 27, 124, 0, 255
3480 DATA 0, 5, 96, 0, 0, 12, 27, 67
3490 DATA 0, 5, 10, 109, 255, 255, 0, 4
3500 DATA 82, 139, 84, 141, 82, 137, 82, 138
3510 DATA 82, 140, 81, 207, 255, 86, 219, 252
3520 DATA 0, 0, 0, 124, 84, 140, 81, 206
3530 DATA 255, 72, 66, 128, 78, 117, 124, 47
3540 DATA 42, 188, 255, 255, 255, 255, 43, 124
3550 DATA 255, 255, 255, 255, 0, 4, 219, 252
3560 DATA 0, 0, 0, 128, 81, 206, 255, 234
3570 DATA 66, 128, 78, 117, 0, 0, 63, 255
3580 DATA 64, 0, 64, 0, 67, 240, 71, 248
3590 DATA 78, 28, 76, 12, 76, 12, 92, 14
3600 DATA 88, 6, 88, 6, 95, 254, 95, 254
3610 DATA 88, 6, 88, 6, 88, 6, 88, 6
3620 DATA 88, 6, 88, 6, 88, 6, 64, 0
3630 DATA 64, 0, 64, 0, 0, 0, 63, 255
3640 DATA 64, 0, 64, 0, 67, 240, 71, 248
3650 DATA 78, 28, 92, 14, 88, 6, 64, 6
3660 DATA 64, 6, 64, 14, 65, 252, 67, 248
3670 DATA 71, 0, 78, 0, 76, 0, 92, 0
3680 DATA 88, 0, 95, 254, 95, 254, 64, 0
3690 DATA 64, 0, 64, 0, 0, 0, 63, 255
3700 DATA 64, 0, 64, 0, 67, 240, 71, 248
3710 DATA 78, 28, 92, 14, 88, 6, 64, 6
3720 DATA 64, 14, 64, 252, 64, 252, 64, 6
3730 DATA 64, 6, 64, 6, 88, 6, 92, 14
3740 DATA 78, 28, 71, 248, 67, 240, 64, 0
3750 DATA 64, 0, 64, 0, 0, 0, 63, 255
3760 DATA 64, 0, 64, 0, 64, 24, 64, 56
3770 DATA 64, 120, 64, 248, 65, 216, 67, 152
3780 DATA 71, 24, 78, 24, 95, 254, 95, 254
3790 DATA 64, 24, 64, 24, 64, 24, 64, 24
3800 DATA 64, 24, 64, 24, 64, 24, 64, 0
3810 DATA 64, 0, 64, 0, 0, 0, 63, 255
3820 DATA 64, 0, 64, 0, 79, 254, 95, 254
3830 DATA 92, 0, 88, 0, 88, 0, 88, 0
3840 DATA 92, 0, 95, 248, 79, 252, 64, 14
3850 DATA 64, 6, 64, 6, 88, 6, 92, 14
3860 DATA 78, 28, 71, 248, 67, 240, 64, 0
3870 DATA 64, 0, 64, 0, 0, 0, 63, 255
3880 DATA 64, 0, 64, 0, 67, 240, 71, 248
3890 DATA 78, 28, 92, 14, 88, 6, 88, 0
3900 DATA 88, 0, 91, 240, 95, 248, 94, 28
3910 DATA 92, 14, 88, 6, 88, 6, 92, 14
3920 DATA 78, 28, 71, 248, 67, 240, 64, 0
3930 DATA 64, 0, 64, 0, 0, 0, 63, 255
3940 DATA 64, 0, 64, 0, 95, 254, 95, 254

```

```

3950 DATA 64, 6, 64, 6, 64, 30, 64, 56
3960 DATA 64, 112, 64, 224, 65, 192, 67, 128
3970 DATA 71, 0, 78, 0, 76, 0, 88, 0
3980 DATA 88, 0, 88, 0, 88, 0, 64, 0
3990 DATA 64, 0, 64, 0, 0, 0, 63, 255
4000 DATA 64, 0, 64, 0, 67, 240, 71, 248
4010 DATA 78, 28, 92, 14, 88, 6, 92, 14
4020 DATA 78, 28, 71, 248, 71, 248, 78, 28
4030 DATA 92, 14, 88, 6, 88, 6, 92, 14
4040 DATA 78, 28, 71, 248, 67, 240, 64, 0
4050 DATA 64, 0, 64, 0, 0, 0, 63, 255
4060 DATA 64, 0, 64, 0, 67, 240, 71, 248
4070 DATA 78, 28, 92, 14, 88, 6, 88, 6
4080 DATA 92, 14, 78, 30, 71, 254, 67, 246
4090 DATA 64, 6, 64, 6, 88, 6, 92, 14
4100 DATA 78, 30, 71, 248, 67, 240, 64, 0
4110 DATA 64, 0, 64, 0, 0, 0, 63, 255
4120 DATA 64, 0, 64, 0, 88, 252, 89, 254
4130 DATA 89, 206, 89, 134, 89, 134, 89, 134
4140 DATA 89, 134, 89, 134, 89, 134, 89, 134
4150 DATA 89, 134, 89, 134, 89, 134, 89, 134
4160 DATA 89, 206, 89, 254, 88, 252, 64, 0
4170 DATA 64, 0, 64, 0, 0, 0, 63, 255
4180 DATA 64, 0, 64, 0, 95, 224, 95, 224
4190 DATA 64, 96, 64, 96, 64, 96, 64, 96
4200 DATA 88, 96, 88, 96, 88, 96, 92, 224
4210 DATA 79, 192, 71, 129, 64, 1, 64, 2
4220 DATA 64, 4, 64, 2, 64, 1, 64, 0
4230 DATA 64, 0, 64, 0, 0, 0, 63, 255
4240 DATA 64, 0, 64, 0, 71, 128, 79, 192
4250 DATA 92, 224, 88, 96, 88, 96, 88, 96
4260 DATA 88, 96, 88, 96, 89, 96, 92, 224
4270 DATA 79, 192, 71, 161, 64, 3, 64, 3
4280 DATA 64, 3, 64, 3, 64, 3, 64, 1
4290 DATA 64, 0, 64, 0, 0, 0, 63, 255
4300 DATA 64, 0, 64, 0, 88, 96, 88, 224
4310 DATA 89, 192, 91, 128, 95, 0, 94, 0
4320 DATA 95, 0, 91, 128, 89, 192, 88, 230
4330 DATA 88, 102, 88, 97, 64, 0, 64, 0
4340 DATA 64, 0, 64, 1, 64, 1, 64, 3
4350 DATA 64, 4, 64, 0, 0, 0, 255, 252
4360 DATA 0, 2, 0, 66, 0, 226, 1, 242
4370 DATA 1, 242, 3, 250, 3, 90, 0, 66
4380 DATA 1, 242, 0, 2, 0, 2, 0, 2
4390 DATA 0, 2, 0, 2, 0, 2, 0, 2
4400 DATA 0, 2, 0, 2, 0, 2, 0, 2
4410 DATA 0, 2, 0, 2, 0, 0, 255, 252
4420 DATA 0, 2, 0, 66, 0, 226, 0, 66
4430 DATA 1, 82, 3, 250, 1, 82, 0, 66
4440 DATA 1, 242, 0, 2, 0, 2, 0, 2
4450 DATA 0, 2, 0, 2, 0, 2, 0, 2
4460 DATA 0, 2, 0, 2, 0, 2, 0, 2
4470 DATA 0, 2, 0, 2, 0, 0, 255, 252
4480 DATA 0, 2, 0, 66, 0, 226, 1, 242
4490 DATA 3, 250, 3, 250, 1, 242, 0, 226
4500 DATA 0, 66, 0, 2, 0, 2, 0, 2
4510 DATA 0, 2, 0, 2, 0, 2, 0, 2
4520 DATA 0, 2, 0, 2, 0, 2, 0, 2
4530 DATA 0, 2, 0, 2, 0, 0, 255, 252
4540 DATA 0, 2, 1, 178, 3, 186, 3, 250
4550 DATA 3, 250, 1, 242, 1, 242, 0, 226
4560 DATA 0, 66, 0, 2, 0, 2, 0, 2
4570 DATA 0, 2, 0, 2, 0, 2, 0, 2
4580 DATA 0, 2, 0, 2, 0, 2, 0, 2
4590 DATA 0, 2, 0, 2, 0, 0, 255, 252
4600 DATA 0, 2, 0, 2, 0, 2, 0, 2
4610 DATA 0, 2, 0, 2, 0, 2, 0, 2
4620 DATA 0, 2, 0, 2, 0, 2, 128, 66
4630 DATA 128, 194, 99, 34, 28, 34, 0, 146
4640 DATA 99, 10, 28, 10, 0, 18, 191, 162
4650 DATA 64, 66, 0, 2, 0, 0, 255, 252
4660 DATA 0, 2, 0, 2, 0, 2, 0, 2
4670 DATA 0, 2, 0, 2, 0, 2, 0, 2
4680 DATA 0, 2, 0, 2, 0, 2, 0, 2
4690 DATA 204, 194, 63, 34, 68, 178, 68, 178
4700 DATA 68, 178, 68, 178, 37, 50, 127, 162
4710 DATA 164, 66, 0, 2, 0, 0, 255, 252
4720 DATA 0, 2, 0, 2, 0, 2, 0, 2
4730 DATA 0, 2, 0, 2, 0, 2, 0, 2
4740 DATA 0, 2, 0, 2, 0, 2, 28, 50
4750 DATA 28, 50, 8, 66, 255, 130, 136, 130
4760 DATA 136, 130, 73, 66, 62, 66, 193, 226
4770 DATA 170, 146, 0, 2, 64, 1, 0, 2
4780 DATA 64, 1, 0, 2, 64, 3, 128, 2
4790 DATA 64, 7, 192, 2, 64, 15, 224, 2
4800 DATA 64, 31, 240, 2, 64, 63, 248, 2
4810 DATA 64, 127, 252, 2, 64, 255, 254, 2
4820 DATA 65, 255, 255, 2, 67, 255, 255, 130
4830 DATA 67, 255, 255, 130, 67, 255, 255, 130
4840 DATA 67, 255, 255, 130, 65, 255, 255, 2
4850 DATA 64, 255, 254, 2, 64, 121, 60, 2

```



```

5120 DATA 64, 0, 0, 2, 63,255,255,252
5130 DATA 0, 0, 0, 0, 64, 96, 12, 2
5140 DATA 65,248, 63, 2, 67,252,127,130
5150 DATA 67,252,127,130, 71,254,255,194
5160 DATA 71,254,255,194, 71,255,255,194
5170 DATA 71,255,255,194, 67,255,255,130
5180 DATA 67,255,255,130, 65,255,255, 2
5190 DATA 64,255,254, 2, 64, 63,248, 2
5200 DATA 64, 31,240, 2, 64, 15,224, 2
5210 DATA 64, 7,192, 2, 64, 3,128, 2
5220 DATA 64, 3,128, 2, 64, 1, 0, 2
5230 DATA 64, 1, 0, 2, 64, 1, 0, 2
5240 DATA 64, 0, 0, 2, 63,255,255,252
5250 DATA 0, 0, 0, 0, 0, 2, 63,255
5260 DATA 255,252, 0, 0, 0, 0, 0, 0
5270 DATA 0, 0, 0, 0, 0, 0, 0, 0
5280 DATA 0, 0, 0, 0, 0, 0, 0, 0
5290 DATA 0, 0, 0, 0, 0, 0, 0, 0
5300 DATA 0, 0, 0, 0, 0, 0, 0, 0
5310 DATA 0, 0, 0, 0, 0, 0, 0, 0
10000 REMARK DATAS TO CHECK YOUR ERRORS
10010 DATA 18597,10364,11388,15197,15999
10020 DATA 11293,18109,15865,12640,12509
10030 DATA 15798,13960,14105,15945,7887
10040 DATA 12190,18252,13556

```

Sinclair/OL World June 1989

	LEA	STOR+12,A3		BSR	ZERO		CMP.B	#200,D1
	MOVEM.L	A4/D6/D4,-(A3)		LEA	BUF+29,A4		BGT	S1AFC
	MOVE.B	#0,-(A4)		MOVEQ	#26,D4		BSR	HORIZ
	MOVE.B	D4,D6		BSR	ZERO		BRA	TROUS4
	BSR	ADDR		LEA	BUF+43,A4	S1AFC	BSR	VERTI
	BSR	DISPLAY		MOVEQ	#12,D4		BRA	TROUS4
	LEA	STOR,A3		BSR	ZERO			
	MOVEM.L	(A3)+,A4/D6/D4		BSR	TROU	HORIZ	LEA	157712,A4
	MOVEQ	#0,D1		BRA	CURSEUR		ADDA.L	D6,A4
	BSR	ADDR					MOVEQ	#7,D2
	BSR	DISPLAY	ZERO	MOVEQ	#12,D2	L2HO	MOVEQ	#3,D3
	LEA	STOR,A3	L1ZE	MOVE.B	(A4),D1	L1HO	MOVE.W	D0,(A4)+
	MOVEM.L	(A3)+,A4/D6/D4		SUBI.B	#1,D1		DBF	D3,L1HO
	SUBI.B	#14,D4		CMP.B	-1(A4),D1		ADDA.L	#120,A4
	BRA	S2AS		BNE	S2ZE		DBF	D2,L2HO
TROUS4	LEA	BUF,A4		ADDQ.L	#1,A4		RTS	
	MOVEQ	#55,D6		SUBI.B	#1,D4			
	MOVEQ	#0,D7		DBF	D2,L1ZE	VERTI	LEA	132218,A4
L1TR4	MOVE.B	(A4)+,D1		RTS			ADDA.L	D7,A4
	CMP1.B	#13,D1	S2ZE	MOVE.B	D4,(A5)+		MOVEQ	#47,D2
	BEQ	S1TR4	L2ZE	MOVE.B	#0,(A4)+	L1VE	MOVE.W	D0,(A4)
	CMP1.B	#26,D1		DBF	D2,L2ZE		ADDA.L	#128,A4
	BEQ	S1TR4		RTS			DBF	D2,L1VE
	CMP1.B	#39,D1	CURSEUR	MOVEQ	#1,D0		RTS	
	BEQ	S1TR4		MOVEQ	#-1,D3			
	CMP1.B	#52,D1		MOVE.L	ID,A0	CUR	DS.L	2
	BEQ	S1TR4		TRAP	#3			
S2TR4	DBF	D6,L1TR4		MOVEQ	#0,D7	EMPLA	DIVU	#8,D6
	CMP1.B	#4,D7		MOVE.W	#-1,D0		DIVU	#6144,D7
	BLT	CURSEUR		LEA	CUR,A5		AND.L	#255,D6
	BRA	BIENP		MOVE.L	(A5),D6		AND.L	#255,D7
S1TR4	MOVEQ	#3,D5		MOVE.W	D6,D7		ADDQ.B	#1,D6
L2TR4	CMP1.B	#0,D6		CLR.W	D6		MULU	#14,D7
	BLE	S2TR4		SWAP	D6		ADD.B	D6,D7
	CMP1.B	#0,(A4)		CMP.B	#244,D1		LEA	SW,A5
	BNE	S2TR4		BNE	S4CU		TST.W	(A5)
	ADDQ.B	#1,D7		LEA	SW,A1		BNE	EMPLPL
	ADDQ.L	#1,A4		TST.W	(A1)	EMPLVI	LEA	BUF,A5
	SUBI.B	#1,D6		BEQ	S5CU		CMP1.B	#0,0(A5,D7)
	DBF	D5,L2TR4		CLR.W	(A1)		BNE	CURSEUR
	BRA	S2TR4		BRA	S4CU		MOVE.B	-1(A5,D7),D1
BIENP	LEA	BUF,A4	S5CU	MOVE.W	#1,(A1)		CMP1.B	#0,D1
	MOVEQ	#3,D5	S4CU	CMP.B	#248,D1		BEQ	CURSEUR
L2BI	MOVEQ	#11,D6		BEQ	BIENP		CMP1.B	#13,D1
L1BI	MOVE.B	(A4)+,D1		CMP.B	#27,D1		BEQ	CURSEUR
	ADDQ.B	#1,D1		BEQ	EXIT		CMP1.B	#26,D1
	CMP.B	(A4),D1		CMP.B	#32,D1		BEQ	CURSEUR
	BNE	QUATRE		BEQ	EMPLA		CMP1.B	#39,D1
	DBF	D6,L1BI		CMP.B	#192,D1		BEQ	CURSEUR
	ADDQ.L	#2,A4		BNE	S1CU		CMP1.B	#52,D1
	DBF	D5,L2BI		CMP.W	#0,D6		BEQ	CURSEUR
	BRA	GAGNE		BEQ	CURSEUR		ADDQ.B	#1,D1
CPT	DS.L	1		BSR	HORIZ		MOVEQ	#55,D6
MESS	DC.B	'ATTEMPT : '	S1CU	SUB.W	#8,D6		SUB.B	D7,D6
				BRA	AFFCU		LEA	STOR+12,A3
QUATRE	MOVEQ	#16,D0		CMP.B	#200,D1		MOVEM.L	A5/D1/D7,-(A3)
	MOVEQ	#17,D1		BNE	S2CU		BSR	ADDR
	MOVEQ	#11,D2		CMP.W	#96,D6		BSR	DISPLAY
	MOVE.L	ID,A0		BEQ	CURSEUR		LEA	STOR,A3
	TRAP	#3		BSR	HORIZ		MOVEM.L	(A3)+,A5/D1/D7
	MOVEQ	#5,D0	S2CU	ADD.W	#8,D6		LEA	BUF,A4
	LEA	CPT,A1		BRA	AFFCU		MOVEQ	#55,D6
	ADDI.L	#1,(A1)		CMP.B	#208,D1	L1EMP	CMP.B	(A4)+,D1
	MOVE.L	(A1),D1		BNE	S3CU		BEQ	TROUVE
	CMP1.B	#4,D1		CMP.W	#0,D7		DBF	D6,L1EMP
	BGE	PERDU		BEQ	CURSEUR	TROUVE	MOVE.B	#0,-(A4)
	ADDI.L	#49,D1		BSR	VERTI		MOVE.B	D1,0(A5,D7)
	MOVEQ	#-1,D3	S3CU	SUB.W	#6144,D7		BSR	ADDR
	MOVE.L	ID,A0		BRA	AFFCU		MOVEQ	#0,D1
	TRAP	#3		CMP.B	#216,D1		BSR	DISPLAY
EFFACE	LEA	TRO,A5		BNE	CURSEUR		BRA	TROUS4
	LEA	BUF+1,A4		CMP.W	#18432,D7			
	MOVEQ	#54,D4		BEQ	CURSEUR	EMPLPL	LEA	BUF,A5
	BSR	ZERO	AFFCU	BSR	VERTI		CMP1.B	#0,0(A5,D7.W)
	LEA	BUF+15,A4		ADD.W	#6144,D7		BEQ	CURSEUR
	MOVEQ	#40,D4		MOVE.W	D8,(A5)+		MOVE.B	0(A5,D7.W),D1
				MOVE.W	D7,(A5)		LEA	BUF,A4
				MOVE.W	#65280,D0		MOVEQ	#54,D6

L1EMPL	SUBI.B	#1,D1		CMP1.B	#14,D7		DBF	D7,L1DIS
	CMP.B	(A4)+,D1		BLT	S1AD5		ADDA.L	#124,A5
SIEMPL	BEQ	SIEMPL		CMP1.B	#28,D7		ADDQ.L	#2,A4
	DBF	D6,L1EMPL		BGE	S2AD5		DBF	D6,L2DIS
SIEMPL	CMP.B	#0,(A4)		SUBI.B	#14,D7		CLR.L	DO
	BNE	CURSEUR		ADDA.L	#6144,A5		RTS	
	ADDQ.B	#1,D1		BRA.S	S1AD5	CLS	MOVEQ	#47,D6
	MOVE.B	#0,0(A5,D7)	S2AD5	CMP1.B	#42,D7	L1CL	MOVE.L	#-1,(A5)
	LEA	STOR,A3		BGE	S3AD5		MOVE.L	#-1,4(A5)
	MOVEM.L	A5/D6/D7,(A3)		SUBI.B	#28,D7		ADDA.L	#128,A5
	MOVEQ	#55,D7		ADDA.L	#12288,A5		DBF	D6,L1CL
	SUB.B	D6,D7		BRA.S	S1AD5		CLR.L	DO
	MOVE.B	D1,0(A5,D7.W)	S3AD5	SUBI.B	#42,D7	UDG	RTS	
	JSR	ADDR		ADDA.L	#18432,A5			
	JSR	DISPLAY	S1AD5	MULS	#8,D7		DC.W	0,16383,16384,16384
	LEA	STOR,A3		ADD.L	D7,A5		DC.W	17392,18424,19996,19468
	MOVEM.L	(A3),A5/D6/D7		RIS			DC.W	19468,23566,22534,22534
	MOVEQ	#0,D1	DISPLAY				DC.W	24574,24574,22534,22534
	MOVEQ	#55,D6					DC.W	22534,22534,22534,22534
	SUB.B	D7,D6		MOVE.B	D1,DO		DC.W	22534,16384,16384,16384
	BSR	ADDR		CLR.L	D1		DC.W	0,16383,16384,16384
	BSR	DISPLAY		MOVE.B	DO,D1		DC.W	17392,18424,19996,23566
	BRA	TROUS4		CLR.L	DO		DC.W	22534,16390,16390,16398
				CMP1.B	#0,D1		DC.W	16892,17400,18176,19968
GAGNE	MOVE.L	ID,A0		BEQ	CLS		DC.W	19456,23552,22528,24574
	MOVEQ	#20,DO		SUBI.B	#1,D1		DC.W	24574,16384,16384,16384
	MOVEQ	#-1,D3		DIVS	#13,D1		DC.W	0,16383,16384,16384
	TRAP	#3		MOVE.W	D1,DO		DC.W	17392,18424,19996,23566
	LEA	MESS6,A1		CLR.W	D1		DC.W	22534,16390,16398,16636
	MOVEQ	#70,D2		SWAP	D1		DC.W	16636,16390,16390,16390
	MOVEQ	#-1,D3		LEA	UDG,A1		DC.W	22534,23566,19996,18424
	MOVEQ	#7,DO		LEA	UDG+824,A2		DC.W	17392,16384,16384,16384
	TRAP	#3		LEA	UDG+816,A3		DC.W	0,16383,16384,16384
	BRA	GETKEY		LEA	UDG+960,A4		DC.W	16408,16440,16504,16632
MESSG	DC.B	10,10,10,10		MOVE.L	D1,D4		DC.W	16856,17304,18200,19992
	DC.B	'		SUBI.L	#10,D4		DC.W	24574,24574,16408,16408
	DC.B	'CONGRATULATIONS		MULS	#48,D4		DC.W	16408,16408,16408,16408
	DC.B	10,10,10		MULS	#48,D1		DC.W	16408,16384,16384,16384
	DC.B	'PRESS'		MULS	#48,DO		DC.W	0,16383,16384,16384
	DC.B	'ANY KEY TO'		MOVE.L	DO,D2		DC.W	20478,24574,23552,22528
	DC.B	'PLAY AGAIN'		ADD.L	DO,DO		DC.W	22528,22528,23552,24568
				MOVEQ	#23,D6		DC.W	20476,16398,16390,16390
PERDU	MOVE.L	ID,A0	L2DIS	MOVEQ	#1,D7		DC.W	22534,23566,19996,18424
	MOVEQ	#20,DO	L1DIS	MOVE.B	0(A1,D1),(A5)		DC.W	17392,16384,16384,16384
	MOVEQ	#-1,D3		MOVE.B	0(A1,D1),1(A5)		DC.W	0,16383,16384,16384
	TRAP	#3		MOVE.B	0(A2,D2),4(A5)		DC.W	17392,18424,19996,23566
	LEA	MESSP,A1		MOVE.B	0(A2,D2),5(A5)		DC.W	22534,22528,22528,23536
	MOVEQ	#70,D2		MOVE.B	0(A4,DO),3072(A5)		DC.W	24568,24092,23566,22534
	MOVEQ	#-1,D3		MOVE.B	0(A4,DO),3073(A5)		DC.W	22534,23566,19996,18424
	MOVEQ	#7,DO		MOVE.B	2(A4,DO),3076(A5)		DC.W	17392,16384,16384,16384
	TRAP	#3		MOVE.B	2(A4,DO),3077(A5)		DC.W	0,16383,16384,16384
	BRA	GETKEY		EORI.W	#65535,(A5)		DC.W	24574,24574,16390,16390
MESSP	DC.B	10,10,10,10		EORI.W	#65535,4(A5)		DC.W	16414,16440,16496,16608
	DC.B	'		EORI.W	#65535,3072(A5)		DC.W	16832,17280,18176,19968
	DC.B	'SORRY BUT YOU		EORI.W	#65535,3076(A5)		DC.W	19456,22528,22528,22528
	DC.B	'FAIL',10,10,10		CMP.W	#144,DO		DC.W	22528,16384,16384,16384
	DC.B	'PRESS'		BLT	S1DIS		DC.W	0,16383,16384,16384
	DC.B	'ANY KEY TO'		MOVE.B	#255,1(A5)		DC.W	17392,18424,19996,23566
	DC.B	'PLAY AGAIN'		MOVE.B	#255,5(A5)		DC.W	22534,23566,19996,18424
			S1DIS	MOVE.B	#255,3073(A5)		DC.W	18424,19996,23566,22534
GETKEY	MOVEQ	#1,DO		MOVE.B	#255,3077(A5)		DC.W	22534,23566,19996,18424
	MOVEQ	#-1,D3		CMP1.W	#0,D4		DC.W	17392,16384,16384,16384
	MOVE.L	ID,A0		BLT	S2DIS		DC.W	0,16383,16384,16384
	TRAP	#3		MOVE.B	4(A5),D3		DC.W	17392,18424,19996,23566
	BRA	INIT		EORI.B	#255,D3		DC.W	22534,22534,23566,19996
				OR.B	0(A3,D4),D3		DC.W	18430,17398,16390,16390
EXIT	MOVE.L	ID,A0		MOVE.B	D3,4(A5)		DC.W	22534,23566,19996,18424
	MOVEQ	#2,DO		CMP.W	#144,DO		DC.W	17392,16384,16384,16384
	TRAP	#2	S3DIS	BLT	S3DIS		DC.W	0,16383,16384,16384
	CLR.L	DO		EORI.B	#255,4(A5)		DC.W	22780,23038,22990,22918
	RTS		S4DIS	MOVE.B	#255,5(A5)		DC.W	22918,22918,22918,22918
				BRA	S4DIS		DC.W	22918,22918,22918,22918
ADDR	MOVEQ	#55,D7		MOVE.B	D3,5(A5)		DC.W	22918,22918,22990,23038
	SUB.B	D6,D7		EORI.W	#65535,4(A5)		DC.W	22780,16384,16384,16384
	LEA	132104,A5		ADDQ.L	#1,A3		DC.W	0,16383,16384,16384
				ADDQ.L	#2,A5		DC.W	24544,24544,16480,16480
				ADDQ.L	#1,A1		DC.W	16480,16480,22624,22624
				ADDQ.L	#1,A2		DC.W	22624,23776,20416,18305
				ADDQ.L	#1,A4		DC.W	16385,16386,16386,16386

DC.W 16385, 16384, 16384, 16384
 DC.W 0, 16383, 16384, 16384
 DC.W 18304, 20416, 23776, 22624
 DC.W 22624, 22624, 22624, 22624
 DC.W 22880, 23776, 20416, 18337
 DC.W 16387, 16387, 16387, 16387
 DC.W 16387, 16385, 16384, 16384
 DC.W 0, 16383, 16384, 16384
 DC.W 22624, 22752, 22976, 23424
 DC.W 24320, 24064, 24320, 23424
 DC.W 22976, 22758, 22630, 22625
 DC.W 16384, 16384, 16384, 16385
 DC.W 16385, 16387, 16388, 16384
 DC.W 0, 65532, 2, 66
 DC.W 226, 498, 498, 1018
 DC.W 858, 66, 498, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 0, 65532, 2, 66
 DC.W 226, 66, 338, 1018
 DC.W 338, 66, 498, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 0, 65532, 2, 66
 DC.W 226, 498, 1018, 1018
 DC.W 498, 226, 66, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 0, 65532, 2, 434
 DC.W 954, 1018, 1018, 498
 DC.W 498, 226, 66, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 0, 65532, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2

DC.W 2, 32834, 32962, 25378
 DC.W 7202, 146, 25354, 7178
 DC.W 18, 49058, 16450, 2
 DC.W 0, 65532, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 7682, 52418, 16162
 DC.W 17586, 17586, 17586, 17586
 DC.W 9522, 32674, 42050, 2
 DC.W 0, 65532, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 2, 2, 2
 DC.W 2, 7218, 7218, 2114
 DC.W 65410, 34946, 34946, 18754
 DC.W 15938, 49634, 43666, 2
 DC.W 16385, 2, 16385, 2
 DC.W 16387, 32770, 16391, 49154
 DC.W 16399, 57346, 16415, 61442
 DC.W 16447, 63490, 16511, 64514
 DC.W 16639, 65026, 16895, 65282
 DC.W 17407, 65410, 17407, 65410
 DC.W 17407, 65410, 17407, 65410
 DC.W 16895, 65282, 16639, 65026
 DC.W 16505, 15362, 16387, 32770
 DC.W 16399, 57346, 16447, 63490
 DC.W 16895, 65282, 16384, 2
 DC.W 16383, 65532, 0, 0
 DC.W 16391, 49154, 16399, 57346
 DC.W 16415, 61442, 16415, 61442
 DC.W 16415, 61442, 16415, 61442
 DC.W 16399, 57346, 16391, 49154
 DC.W 16865, 3842, 17393, 8066
 DC.W 18425, 16322, 18431, 65474
 DC.W 18431, 65474, 18427, 49090
 DC.W 17393, 8066, 16865, 3842
 DC.W 16385, 2, 16387, 32770
 DC.W 16399, 57346, 16447, 61442
 DC.W 16895, 65282, 16384, 2
 DC.W 16383, 65532, 0, 0
 DC.W 16385, 2, 16385, 2

DC.W 16387, 32770, 16387, 32770
 DC.W 16391, 49154, 16391, 49154
 DC.W 16399, 57346, 16415, 61442
 DC.W 16511, 64514, 16895, 65282
 DC.W 18431, 65474, 16895, 65282
 DC.W 16511, 64514, 16415, 61442
 DC.W 16399, 57346, 16391, 49154
 DC.W 16391, 49154, 16387, 32770
 DC.W 16387, 32770, 16385, 2
 DC.W 16385, 2, 16384, 2
 DC.W 16383, 65532, 0, 0
 DC.W 16480, 3074, 16888, 16130
 DC.W 17404, 32642, 17404, 32642
 DC.W 18430, 65474, 18430, 65474
 DC.W 18431, 65474, 18431, 65474
 DC.W 17407, 65410, 17407, 65410
 DC.W 16895, 65282, 16639, 65026
 DC.W 16447, 63490, 16415, 61442
 DC.W 16399, 57346, 16391, 49154
 DC.W 16387, 32770, 16387, 32770
 DC.W 16385, 2, 16385, 2
 DC.W 16385, 2, 16384, 2
 DC.W 16383, 65532, 0, 0
 DC.W 2, 16383, 65532, 0
 DC.W 0, 0, 0, 0



EEC LTD QL from £85 with £30 extra* software FREE!

Complete in original boxes, as new, JS Rom — Latest version S'Ware
 Quill, Abacus, Archive & Easel, Comprehensive Bound User Guide For
 QL, Superbasic & Above Programs (90 Day Warranty) **£140***

As above but factory reconditioned with basic QL & Quill Guide JS ROM **£120***
 Same Pack But With JM ROM **£110***
 Reconditioned QL Only JM ROM **£85***

*FREE SOFTWARE

HOME FINANCE and TOUCH & GO or, DECISION MAKER with *items
 (while stocks last)

QL SPARES ETC.

Basic working P/C BOARDS complete but without Microdrives **£50**

Comprehensive Bound User Guide including Quill, Abacus,
 Archive and Easel instructions **£15**

Microdrive Units — £20 JM ROMS — £15 QL Power Supplies — £15
 QL Centronic Interface — £19.95 (Both include Leads) Joystick — £12.95

QL SELECTED LOW COST PRINTERS WITH I/FACE & LEADS

SEIKOSHA GP100A 80 Col., 50 CPS Centronics, Tractor Feed, Graphics, etc. **£95***
 (send SAE for CL printouts)

SEIKOSHA GP550A 50 CPS, 25 NLQ low noise, 140 characters, Friction/
 Tractor Feed (send SAE for CL printouts) **£129***

SOFTWARE CLEARANCE

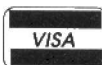
SINCLAIR 4 PACK — Quill - Archive - Easel - Abacus - Latest issues **£19.95**
 CASH TRADER — Comprehensive business records and graphics **£24.95**
 DECISION MAKER — For business and domestic problem evaluation **£24.95**
 HOME FINANCE — Keeps track of your bank accounts and personal expenditure **£24.95**
 QL PAINT Superb graphics, QL Touch & Go, Typing Tutor, QL Gardener **each £14.95**

Terms CWO - ACCESS or VISA
 Minimum order £10.00, p & p £6.00, Printers and QL
 Other items £3.00. Overseas enquire.
 Fax No: 0753-887149

Orders to:

EEC LTD

18-21 Misbourne House, Chiltern Hill, Chalfont St. Peter, Bucks. SL9 9UE
 Tel: 0753 888866



XVI Version 4

Version 4 of THOR XVI is a completely redesigned hardware and software version of the well-known CST THOR XVI, now a 100% Danish production. It is produced by one of the largest Scandinavian manufacturers of high quality measuring instruments vouching for the high standard of our product.

The brand new version of ARGOS (6.40/1.07) introduces a number of new facilities and a much improved full SCSI hard disc handling. ARGOS 6.40 now supports the standard IBM AT keyboards, IBM AT extended and PS/2 extended keyboards, covering 11 different national languages.

Now it is possible to use the advanced facilities of any printer, including laserprinters, thanks to new user defined extended translation tables.

ARGOS winding facilities are extended to support separate screen MODE for each job.

In addition to PSION Xchange we deliver free of charge 2 discs with easy to use menu system and utilities.

Present THOR XVI users please register with us in order to request a ROM/Software upgrade. Technical support is already available in the U.K. through P.M. Engineering.

Please request the new 12 pages THOR XVI brochure and price list containing also a description of the new coming products. We are pleased to announce 2 new products:

★ **ARCTURUS EDITOR (Arced)**, the indispensable companion for programmers (also running on THOR 8 and Sinclair QL), introductory price: **GBP £45.00**. Please request a technical description.

★ **C++ COMPILER** for the THOR and QL range of products, is under development.

Dansoft offers software service contracts for existing THOR customers

DANSOFT

Sole Agent for:

THOR INTERNATIONAL COMPUTER SYSTEMS I/S

Raadhustraede 4 b, 4.sal, DK 1466 Copenhagen K
 Mail to: P.O. Box 59, DK 1002 Copenhagen K, Denmark.
 Phone no: +45 1 9390305 (after May 15th: +45 33930305)
 Fax no: +45 1 938292 (after May 15th: +45 33 938292)



QL SUB

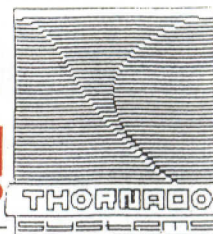
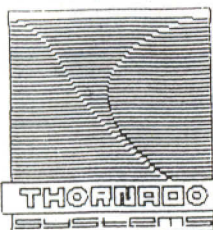


Telephone: 0388-450610, Fax: 0388-609845, Prestel MBX: 219998590.
Telex: 934999 TXLINK G (Quoting Reference No. 219998590)

0388 450610 THE QL'S 0388 450610

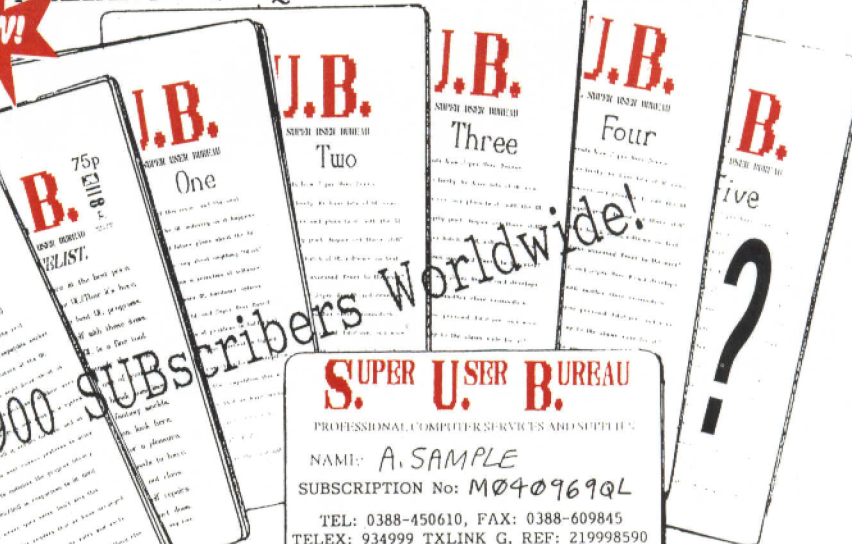
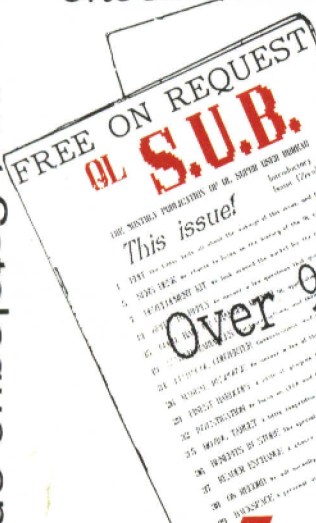
REVOLUTION HAS ARRIVED!

QL SUB, THORNADO, QLCF, AND MORE-
ALLIANCE OF QL USER GROUPS!



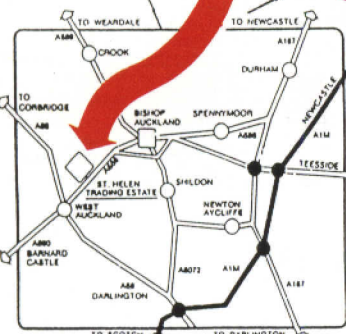
NEW!

Thorus



S. U. B.
PROFESSIONAL COMPUTER SERVICES AND SUPPORT
NAME: A. SAMPLE
SUBSCRIPTION No: M040969QL
TEL: 0388-450610, FAX: 0388-609845
TELEX: 934999 TXLINK G, REF: 219998590
PRESTEL MBX: 219998590
PO BOX 3, SHILDON, DL4 2LW

SUBSCRIPTION OFFER!



What do we offer?

A light at the end of the tunnel...
A full-time telephone help-line is backed up by written support, and a wide range of special services developed only for our readers. 12 issues of "QL SUB" magazine at (we hope) monthly intervals. Savings on hardware/software. Unbiased and informed advice.

Richard J. Turner.

Richard J. Turner,
(Editor, "QL SUB")

Who can benefit?

All sorts of people read "QL SUB" in all sorts of places- Papua New Guinea, Iran AND Iraq, Australia, United States, India, France, Oman, Zimbabwe, Libya, Greece, Poland...

£15 (UK ONLY)
USUAL RATE STARTS £20

£20 Europe, £25 World.

OFFER ENDS 1st JULY

PHONE US NOW
FOR BEST PRICES

Auckland Business Centre, St. Helen's Auckland,
BISHOP AUCKLAND, Co. Durham, ENGLAND, DL14 9TX.

FRIENDLY ADVICE
ON COMPUTERS

PO Box 3, Shildon, DL4 2LW (UK)

Please Note- We use PO BOX 3 for convenience. If you would prefer to write to, or order from our business premises direct you are welcome to do so. Of course, you can call too, but please arrange a time to call.

WE WELCOME CALLERS BY APPOINTMENT ONLY!

QL Bulletin Board always on 0388-773737
Running Every Day. All Day. Direct Dial!

FREE Magazine and Catalogue on Request!
Just Ask for Information. No Obligation.